8-20-2013

Wavelet Neural Network Based Very Short-term Load Forecasting and Prediction Interval Estimation

Che Guan
University of Connecticut - Storrs, che.guan@engr.uconn.edu

Wavelet Neural Network Based Very Short-term Load Forecasting and Prediction Interval Estimation

Che Guan, Ph.D.

University of Connecticut, 2013

Very short-term load forecasting predicts the loads in electrical power network one or several hours into the future in steps of a few minutes (e.g., five minutes) in a moving window manner based on online data collected every few seconds (e.g., four seconds). In order to quantify forecasting accuracy in real-time, the forecasting process should also estimate good prediction intervals online. Accurate forecasting with prediction intervals is important for resource dispatch and area generation control, and helps power market participants make prudent decisions. It is, however, difficult in view of the noisy data collection process with possible malfunctioning of data gathering devices, the different characteristics of load frequency components, and the accurate derivation and evaluation for prediction interval estimation in real-time.

This thesis presents a method of multilevel wavelet neural networks with data pre-filtering. The key idea is to use a spike filtering technique to detect spikes in load and correct them without altering the normal load. Wavelet decomposition is then used to decompose the load into multiple components at different frequencies, separate neural networks are applied to capture the features of individual components, and results of neural networks are

then combined to form the final forecast. To perform moving forecast over an hour, twelve dedicated structures are used based on testing results.

Because wavelet neural networks are based on back propagation without estimating prediction intervals, the method is extended by using hybrid Kalman filters to produce forecasting with prediction interval estimates online. Based on data analysis, a neural network trained by an extended Kalman filter is used for the low-low frequency component to capture the near-linear relationship between the input load component and the output measurement, while neural networks trained by unscented Kalman filters are used for low-high and high frequency components to capture their nonlinear relationships. The overall variance estimate is then derived and evaluated for prediction interval estimation.

Testing results demonstrate the effects of data pre-filtering, the accuracy of wavelet neural networks, the effectiveness of hybrid Kalman filters for capturing different features of load components, and the accuracy of derived prediction interval estimates, based on a data set from ISO New England.

Wavelet Neural Network Based Very Short-term Load Forecasting and Prediction Interval Estimation

Che Guan

B.S., Changchun University of Science and Technology, Changchun, China, 2004
M.S., Chinese Academy of Sciences, Beijing, China, 2007
M.S., University of Connecticut, Storrs, USA, 2011

A Dissertation

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

At the

University of Connecticut

2013

APPROVAL PAGE

Doctor of Philosophy Dissertation

Wavelet Neural Network Based Very Short-term Load Forecasting and Prediction Interval Estimation

Presented by

Che Guan, B.S., M.S.

Major Advisor: _____

Peter B. Luh

Associate Advisor: _____

Yaakov Bar-Shalom

Associate Advisor: _____

Laurent D. Michel

University of Connecticut

2013

To My Parents

Acknowledgements

It has been a great pleasure to spend six years as a graduate student at UConn. Here I met many wonderful scientists and friends who play important roles on me to become a researcher, and have contributed significantly to the work in my dissertation.

I would like to express my sincerest gratitude to my advisor, Professor Peter B. Luh, for his continuous guidance, encouragement, patience, and opportunities given to me to practice and grow up. All the works could not be accomplished without his persistent supervision, and I am appreciated very much. I am also very grateful to Dr. Ying Chen. As a graduated member from the lab as well as a best friend, she has been selflessly sharing her precious experiences and helping me overcome many difficulties on the way to the Ph.D. target. It is really a fortunate to work with her.

I would like to thank my co-advisors Professor Yaakov Bar-Shalom and Professor Laurent Michel for their supervision and help on my research. I am also thankful to Professor Zhiyi Chi for his consistent help on my research, and Professor Shengli Zhou for his review of this dissertation. I would also like to thank Dr. Kwok Cheung at the Alstom Grid, and Peter B. Friedland who was at the ISO New England for their continuous supports on my research, as well as Mr. Matthew A. Coolbeth, Mr. Yuting Wang, and Mr. Stephen Corbo, for their continuous supports on my research. I am also thankful to Miss Fang Chen, Dr. Yi Yang, Dr. Tian Mi, Dr. Shuo Zhang, Dr. Ying Wang, etc., for their information and consistent help on my dissertation.

I would like to thank all the past and present lab members at Manufacturing Systems Laboratory: Dr. Joseph Yan, Dr. Xing Zhao, Dr. Li Zhang, Dr. Feng Zhao, Dr. Ying Chen,

Dr. Jin Sun, Dr. Guoyu Tu, Dr. Mingyang Li, Mr. William E. Blankson, Mr. Abhinaya Joshi, Mr. Majid Chauhdry, Mr. Peng Wang, Mrs. Bingjie Zhang, Miss Congcong Wang, Mr. Xu Han, Mr. Liangliang Sun, Mr. Yige Zhao, Mrs. Weihua Wang, Mrs. Bing Yan, Mr. Mikhail Bragin, Mrs. Yu Chen, Mr. Ying Yan, Mr. Yaowen Yu, Miss Xiaorong Sun, Mr. Weiji Han, Mr. Biao Sun, Mr. Christian Wilkie, etc. It was a great pleasure to meet you, and I've enjoyed working with you and take tremendous pride in what we've been able to accomplish. The six-year life in the lab leaves with me many fond memories and valued relationships.

I would like to express my sincere appreciation for my colleagues at the Dun and Bradstreet, especially for: Dr. Nipa Basu and the whole worldwide predictive analytic team's consistent supports, and Dr. Anthony Scriffignano for his encouragement, help, and support on my oral defense of dissertation.

Finally I would like to thank my family, especially for my parents Mr. Yongjun Guan and Mrs. Jun Sun. Whenever, wherever, and whatever I am in need, you are always there. Without your support, I would have not been able to accomplish as much as I have.

This work was supported in part by ISO New England and Alstom Grid.

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Research Motivation

Very short-term load forecasting (VSTLF) predicts the loads in electrical power network one or several hours into the future in steps of a few minutes (e.g., five minutes) in a moving window manner based on online data collected every few seconds (e.g., four seconds). To quantify forecasting accuracy in real-time, the forecasting process should also estimate accurate prediction intervals (PI) online. Accurate VSTLF with good PIs is important for resource dispatch and area generation control, and helps power market participants make prudent decisions. Based on data analysis, load time series have multiple frequency components, and each may have its unique pattern, such as monthly, weekly, and hourly patterns. Effective VSTLF, however, is difficult in view of the noisy data collection process with possible malfunctioning of data gathering devices, different characteristics of load components, and the accurate derivation for estimating prediction intervals online.

1.2 Dissertation Outline

The research of this dissertation is to advance real-time load forecasting methods in electrical power network. The study is an extension of the previous method for short-term load forecasting (STLF) which predicts the loads of tomorrow in hourly steps based on the single-level wavelet decomposition and neural networks trained through using a data set from ISO New England (Chen et al., 2010). The method presented a way for handling load features at different frequencies. However, the load features of STLF are quite different from

the ones of VSTLF because short-term load data have fewer patterns than very short-term load data to be analyzed. Also, spikes were not considered because they had been removed by ISO New England before STLF was performed, whereas removing spikes is a critical issue for VSTLF.

In Chapter 2, wavelet neural networks (WNN) with data pre-filtering will be developed to forecast the loads one hour into the future in five-minute steps in a moving window manner. To effectively remove spikes, it is observed that spikes may have different magnitudes and widths. Thus, they are classified into micro and macro spikes at either four-second or five-minute resolutions. Micro and macro filtering techniques will be developed to effectively detect and filter them out. To accurately capture load features, the wavelet technique will be used to decompose the loads into multiple frequency components. Each component is then appropriately transformed, normalized, and fed with time and date indices to a neural network, so that the features of individual components are properly captured. Forecasts from individual neural networks are then transformed back and combined to form the final forecasts. To perform moving forecasts, twelve dedicated wavelet neural networks will be used based on preliminary simulation results.

In Chapter 3, the method of wavelet neural networks will be further improved to provide prediction intervals. By replacing the first-order back propagation algorithm with second-order Kalman type algorithms, dynamic covariance can be produced for prediction interval estimation. The method of wavelet neural networks trained by hybrid Kalman filters (WNNHK) will be developed. It forecasts the loads one hour into the future in 5-min steps in a moving window manner with associated PI estimates online. After a data analysis, it is found that the Low-Low (LL) frequency component has a near-linear relationship between

the low frequency load input and its measurement, whereas the Low-High (LH) and High (H) frequency components have the nonlinear relations. To capture the near-linear relationship between the input and measurement for the LL component, the extended Kalman filter is used to train a neural network (EKFNN) because the extended Kalman filter is derived by linearizing the system and is good for near-linear systems. To capture highly nonlinear relationships for the LH and H components, the unscented Kalman filter is used to train neural networks (UKFNN) because the unscented Kalman filter is good for highly nonlinear systems. To accurately estimate prediction intervals online, the overall variance estimate will be calculated by summing up the three orthogonal variance estimates from H, LH, and LL frequency neural networks. The estimates for H and LH components are directly obtained. The estimate for the LL component is further derived because the relative increment, a nonlinear transformation, is applied to the LL component. The relative increment is used to make the series stationary so that the transformed series can be easily captured.

All the works are implemented in MATLAB, and configured through training, validation, and test data sets. The open source code and the part of the test data and results are open, and can be obtained from http://github.com/ldmbouge/vstlf. The software was run on a server with dual Xeon quad core Intel E5620 2.4GHz processors and a 36 GB memory. Testing results will demonstrate the values of data pre-filtering, wavelet decomposition, load transformation, neural networks, and dedicated wavelet neural networks for VSTLF. The results will also illustrate the effectiveness of hybrid Kalman filters for capturing different features of load components, and the accuracy of the overall variance estimate derived based on a data set from ISO New England.

1.3 Major Contributions

Spikes are analyzed with respect to magnitudes and widths, and then classified at either 4 second or 5 minute resolutions. Micro and macro filtering techniques are further developed to effectively filter spikes out.

Amplitude spectrum shows loads have several components. A wavelet method is selected to separate the load into proper levels. Parameters in wavelet transform are discussed, derived, and selected. To help capture load features, each component is properly transformed and fed with time and date indices to an NN. Finally, twelve dedicated WNNs are used for moving forecasts.

To produce prediction interval estimate online, hybrid Kalman filters are developed to train wavelet neural networks and to capture the complicated load features.

Based on data analysis, an NN trained by an extended Kalman filter is used to capture the near-linear relation between the Low-Low input and output measurement, whereas NNs trained by unscented Kalman filters are developed to capture highly nonlinear relations for Low-High and High frequency components.

Due to the nonlinear transformation for load inputs, the overall variance is further derived. The distribution of the forecasting errors is analyzed, and prediction interval estimates are thoroughly evaluated in several ways.

1.4 Publications

Journals:

[1] Y. Chen, P. B. Luh, C. Guan, Y. Zhao, L. D. Michel, M. A. Coolbeth, P. B. Friedland, and S. J. Rourke, "Short-Term Load Forecasting: Similar Day-Based Wavelet Neural

Networks," *IEEE Transactions on Power Systems*, Vol. 25, No. 1, pp. 322-330, February 2010.

[2] C. Guan, P. B. Luh, L. D. Michel, Y. Wang, and P. B. Friedland, "Very Short-term Load Forecasting: Wavelet Neural Networks with Data Pre-filtering," *IEEE Transactions on Power Systems*, Vol. 28, No. 1, pp 30-41, February 2013.

[3] C. Guan, P. B. Luh, L. D. Michel, and Z. Chi, "Hybrid Kalman Filters for Very Short-term Load Forecasting and Prediction Interval Estimation," *IEEE Transactions on Power Systems*, to appear.

Conference Proceedings:

[4] C. Guan, P. B. Luh, M. A. Coolbeth, Y. Zhao, L. D. Michel, Y. Chen, C. J. Manville, P. B. Friedland, and S. J. Rourke, "Very short-term load forecasting: Multilevel wavelet neural networks with data pre-filtering," *Proceedings of the IEEE Power and Energy Society 2009 General Meeting*, Calgary, Alberta, Canada, July 2009.

[5] C. Guan, P. B. Luh, L. D. Michel, Y. Bar-Shalom, and P. B. Friedland, "Interacting Multiple Model Approach for Very Short-Term Load Forecasting and Confidence Interval Estimation," *Proceedings of the 8th World Congress on Intelligent Control and Automation*, Jinan, Shandong, China, June 2010.

[6] P. B. Luh, L. D. Michel, P. B. Friedland, C. Guan, and Y. Wang, "Load Forecasting and Demand Response," *Proceedings of the IEEE Power and Energy Society 2010 General Meeting*, Minneapolis, Minnesota, July 2010.

[7] C. Guan, P. B. Luh, L. D. Michel, M. A. Coolbeth, and P. B. Friedland, "Hybrid Kalman algorithms for very short-term load forecasting and confidence interval

estimation," *Proceedings of the IEEE Power and Energy Society 2010 General Meeting*, Minneapolis, Minnesota, July 2010.

[8] C. Guan, P. B. Luh, and W. Cao, "Short-term Wind Generation Forecasting and Confidence Interval Estimation Based on Neural Networks Trained by Extended Kalman Particle Filter," *Proceedings of the 9th World Congress on Intelligent Control and Automation*, Taipei, Taiwan, June 2011.

[9] C. Guan, P. B. Luh, W. Cao, L. D. Michel, and K. W. Cheung, "Dual-tree M-band Wavelet Transform and Composite Very Short-term Load Forecasting," *Proceedings of the IEEE Power and Energy Society 2011 General Meeting*, Detroit, Michigan, July 2011.

2. Very Short-term Load Forecasting: Wavelet Neural Networks with Data Pre-filtering

2.1 Introduction

Very short-term load forecasting predicts the loads one or several hours into the future in steps of a few minutes (e.g., five minutes) in a moving window manner based on online data collected every few seconds (e.g., four seconds). Accurate load forecasting has traditionally been important since it is critical for automatic generation control and resource dispatch, and it also ensures revenue adequacy for the Independent System Operator (ISO) multi-settlement markets. Effective VSTLF, however, is difficult in view of the noisy data collection process with possible malfunctioning of data gathering devices and complicated load features.

Methods for very short-term load forecasting are limited. Existing methods of persistence, extrapolation, time series, Kalman filtering, fuzzy logic, and neural networks (NN) will be reviewed in Section 2.2. Among these methods, neural networks have been widely used. A standard NN was used for VSTLF (Liu et al., 1996). To improve data stationarity, inputs to an NN were transformed by using logarithmic differences in (Shamsollahi et al., 2001) and by using relative increments in (Charytoniuk and Chen, 2000). A single neural network, however, may not be able to accurately capture complicated load features because the load data have multiple frequency components, and each may have its unique pattern. Furthermore, spikes are randomly distributed over time and have different magnitudes and widths. They affect neural network training, and result in degraded predictions. An intuitive way to filter the spike is to compare the

measured and predicted loads, and if the absolute value of the difference is greater than a threshold, a spike is said to be detected. The spike was then replaced by the interpolated value (Shamsollahi et al., 2001) or the predicted value (Xie et al., 1996). This way, however, may not be effective. To reduce the effects of spikes, further analysis and filtering are needed.

Recently, we have developed a method for short-term load forecasting (STLF) which predicts the loads of tomorrow in hourly steps based on the single-level wavelet decomposition and neural networks trained through using a data set from ISO New England (Chen et al., 2010). A correction coefficient scheme was also developed to enhance predictions around holidays (Zhao et al., 2009). These methods presented a way for handling load features at different frequencies. However, the load features of STLF are quite different from the ones of VSTLF because short-term load data have fewer patterns than very short-term load data to be analyzed in Subsection 2.4.1. Also, spikes were not considered because they had been removed by ISO New England before STLF was performed, whereas removing spikes is a critical issue for VSTLF.

In this chapter, wavelet neural networks (WNN) with data pre-filtering are developed to forecast the loads one hour into the future in five-minute steps in a moving window manner. To effectively remove spikes, it is observed that spikes may have different magnitudes and widths. Thus, they are classified into micro and macro spikes at either four-second or five-minute resolutions. Micro and macro filtering techniques are developed in Section 2.3 to effectively detect and filter them out. The advantage of filtering spikes in the four-second data series is to provide the leading indicator to the

operator of a potential SCADA telemetry problem in real-time. Filtering spikes in the five-minute data series is often a lagging indicator of faulty load telemetry.

Wavelet neural networks are developed in Section 2.4. The wavelet technique is used to decompose the loads into multiple frequency components. Each component is then appropriately transformed, normalized, and fed with time and date indices to a neural network, so that the features of individual components are properly captured. Forecasts from individual neural networks are then transformed back and combined to form the final forecasts. To perform moving forecasts, twelve dedicated wavelet neural networks are used based on test results.

In Section 2.5, the method is configured through training, validation, and test data sets as presented in (Ripley, 1996: Chapter 2). Example 1 uses a classroom-type problem to illustrate the effects of the wavelet decomposition. Based on the data set from ISO New England (ISO-NE), Example 2 demonstrates the values of data pre-filtering, wavelet decomposition, load transformation, neural networks, and dedicated wavelet neural networks for VSTLF. The code as well as part of the test data and results are open, and can be downloaded at http://github.com/ldmbouge/vstlf.

2.2 Literature Review

Not many papers report the handling of spikes. One way is to compare measured and predicted loads, and if the absolute values of the differences are greater than a threshold, spikes are declared and then replaced by predicted values in (Xie et al., 1996). Another way is to replace observed spikes by zeros which are then fixed by using a splining algorithm. If the length of zeros is too long, interpolations from a similar day's

loads are used to fill zero-valued data (Shamsollahi et al., 2001). These methods are valuable. However, they are prone to errors due to the uncertain nature of the load data and the various magnitudes and widths of spikes. Spikes replaced by bad values may degrade future predictions. Therefore, spikes have to be further analyzed, and effective ways are highly needed for filtering them out.

Spike filtering has also been reported for short-term load forecasting. In comparison to VSTLF, spikes in STLF have different features with respect to magnitudes and widths because of the integrative nature of short-term load data and the fact that most spikes should have been removed before STLF is performed. The simple techniques consisting of if-then rules, low pass filtering, and NN based self-filtering were used to handle STLF spikes in (Fidalgo and Peças Lopes, 2005). Recently, entropy related functions, which are robust to noisy data, were developed in (Liu et al., 2007) and were further applied to the training of neural networks for future three-day wind power forecasting (Bessa et al., 2009). To perform the online training, a self-adaptive approach was used in (Bessa et al., 2009), where "the information potential of the error" was recursively estimated. Although these methods are robust to noisy data, in order to help a forecasting model learn normal load patterns rather than complicated noisy data in real-time, it is desirable to remove spikes before data are used for VSTLF.

Limited VSTLF methods have been reported in the literature, and they include methods of persistence, extrapolation, time series, fuzzy logic, Kalman filtering, and neural networks. Persistence forecasting (Fox et al., 2007) may be the simplest method, and it assumes that the forecast data will be the same as the last measured values. This is not sufficient for VSTLF because very short-term load series change in real-time.

Extrapolation predicts the load based on the past by using a least square algorithm (Wang et al., 1996) or by using a curve fitting algorithm based on a shape similarity criterion (Luo and He, 2007). The load increment was predicted through a weighted average of increments of previous loads in (Zhou et al., 2005). A dynamic clustering method was used to pre-group the loads into multiple groups, and load increments were then forecasted in (Yang et al., 2005).

Similar to the extrapolation method, the auto-regression method uses a simple linear combination of the previous load series for prediction(s). Its coefficients were tuned on-line using the least mean square algorithm in (Liu et al., 1996). The method was extended to autoregressive integrated moving average (ARIMA) for load forecasting, and parameters were updated via a recursive least square algorithm with a forgetting factor in (Lu et al., 2005). ARIMA was extended to seasonal autoregressive integrated moving average to capture the seasonal load feature in (De Andrade and Da Silva, 2010). Support vector regression method was developed for VSTLF, which was used with kernel functions to create complex nonlinear decision boundaries in (Setiawan et al., 2009). Holt-Winters adaptation and the new intraday cycle exponential smoothing method were used together for predictions in (Taylor, 2008).

Kalman filter was applied to VSTLF in a few references. For example, the loads were separated into deterministic and stochastic components, and both were predicted via Kalman filters in (Trudnowski and Mcreynolds, 2001). While in (Xie et al., 1996), the deterministic and stochastic components were predicted via the least square algorithm and Kalman filter, respectively. Fuzzy logic methods convert input data to fuzzy values which are then compared with patterns extracted from the training process. The most

similar fuzzy value was chosen and then mapped to the prediction in (Liu et al., 1996). Fuzzy logic was also combined with neural networks to form a fuzzy neuron system, and the parameters of which were configured via chaotic dynamics reconstruction techniques in (Yang et al., 2006; Kawauchi et al., 2004). A hybrid neuron-fuzzy approach was developed in (de Andrade and da Silva, 2010), which used the cross validation methodology to choose inputs, membership functions, and optimization methods.

Among all these VSTLF methods, neural networks have been widely used. They assume a nonlinear functional relationship between the loads to be forecasted and affecting factors, and estimate the weights based on historical data. Their inputs may include the time and date indices, the loads of previous hour, and the loads of yesterday and last week with the same time and date indices to the forecasting hour. For example, different feature sets of historical load data were tested in (Koprinska et al., 2010). Weather information is seldom used for VSTLF due to the large time constant of the load (Charytoniuk and Chen, 2000). Transformations of load inputs, e.g., the logarithmic difference and relative increment, have been reported to improve data stationarity in (Shamsollahi et al., 2001; Charytoniuk and Chen, 2000). Also, different neural networks were used for different periods of a day in (Charytoniuk and Chen, 2000). These neural network methods provide valuable information for the input selection and transformation. However, very short-term load data have complicated features, and few papers present thorough analysis and effective ways to capture load features in real-time.

2.3 Data Pre-filtering

For ISO New England, load data are collected from data collecting devices every four seconds and then aggregated into five-minute loads. Because of possible malfunctioning of collecting devices, spikes exist within load data. These spikes do not reflect true loads and, as a result, affect NN training and degrade predictions. Potential spikes are observed having varying magnitudes and widths at either four-second or five-minute resolutions, and they are randomly distributed over time. A spike is said to be detected if the absolute value of the difference between the original load and this smoothed load exceeds a threshold. Spikes are then classified as "micro spikes" and "macro spikes" based on their widths. A micro spike is defined if its width is smaller than a threshold w_1 (in terms of number of resolution units of either four seconds or five minutes), whereas a macro spike is defined if its width is in-between two thresholds w_1 and w_2. These thresholds are determined based on training, validation, and test data sets. It is difficult to differentiate the spikes with widths larger than w_2 from regular load changes. However, this situation usually requires human intervention, and will not be considered.

To filter these spikes, micro spikes are first recognized at the four-second resolution and are removed by using the micro spike filtering method which will be presented in Subsection 2.3.1. After aggregating into five-minute loads, micro spikes are again recognized and filtered at the five-minute resolution by using the same method. Finally, macro spikes are recognized and processed at the five-minute resolution by using the macro spike filtering method which will be presented in Subsection 2.3.2. Macro spike

filtering is only applied to the loads at the five-minute resolution because macro spikes at the four-second resolution may become micro spikes after integration.

2.3.1 Micro Spike Filtering

The key idea for filtering the micro spike is the use of a zero phase filter to obtain the smoothed load. If the absolute value of the difference between the original load and this smoothed load exceeds a threshold, a spike is said to be detected. Then, the spike is replaced by the smoothed load. This method is first applied to the loads at the four-second time resolution and then at the five-minute time resolution.

Intuitively, the response of a zero phase filter to a rectangular pulse function should be a smoothed and symmetric function without shifting in time. This filter is realized by a unit impulse response symmetric with respect to the time zero axis. When taking Fourier transform, the resulting function should have the phase identically equal to zero. Such a filter is called a zero phase filter as described in (Smith, 1999: Chapter 19). In practice, the idea is to take the average of the actual data in the time-forward and reversed operations with equal weights over the filter window as explained below (Mitra, 2006: pp. 604-605). The result from the zero phase filter has precisely zero phase distortion and magnitude modified. Let the input sequence at time $t+N$ be denoted as X = $\{x(t+1), \ldots, x(t+N)\}$, where N is the length of the latest load inputs to be processed in real-time. Sequence Y = $\{y(t+w), \ldots, y(t+N)\}$ is sequentially produced by the filter with the width w in the following time-forward operation of the zero phase filter:

$$y(t+n) = \sum_{i=n-w+1}^{n} x(t+i)/w, \quad n = w, \cdots, N . \qquad (1)$$

The above sequence is appended by $\{x(t+N+1), \ldots, x(t+N+w-1)\}$ (explained in the next paragraph) to make sure the sequence $\{y(t+N-w+1), \ldots, y(t+N)\}$ after the time-reversed operation of the zero phase filter has a similar magnitude to the load segment $\{y(t+w), \ldots, y(t+N-w)\}$. Following (Mitra, 2006), the resulting sequence is reversed and run through the same filter again. The output of this second filtering is then time reversed to generate the final smoothed series $Z = \{z(t+w), \ldots, z(t+N)\}$.

Since a zero phase filter is causal, the load inputs have to be appended. The load segment $\{x(t+N+1), \ldots, x(t+N+w-1)\}$ is available during training. However, it is not available during real-time forecasting. To append load inputs with a reasonable sequence, the load segment $\{x(t+N-w+1), \ldots, x(t+N)\}$ is mirrored horizontally and flipped vertically with respect to the point $(t+N, x(t+N))$ in the coordinate space. Based on observation, this is because the changes of load series over a short time period have similar slopes for most of the times.

To detect spikes, a sequence of the difference $D = \{d(t+w), \ldots, d(t+N)\}$ between the smoothed and actual is obtained by:

$$d(t+n) = z(t+n) - x(t+n), \quad n = w, \cdots, N . \qquad (2)$$

A micro spike is said to be detected if the absolute value of $d(t+n)$ exceeds a threshold m, and the width of the spike is smaller than a threshold w_1 (the width of the processing window). To replace a spike with a corrected signal, the value of $x(t+n)$ is replaced with $z(t+n)$. These thresholds are analyzed and then determined through training, validation, and test processes in a three way data split.

Figure 2-1 depicts the four-second load series before and after the micro spike filter is applied. The spikes with widths smaller than the processing window w_1 (micro spikes,

as marked by the three small red circles) are removed by the filter. Spikes with widths close to or greater than the processing window w_1 (macro spikes, as marked by the large black ellipse) are only attenuated or cannot be handled by the micro spike filter at the four-second time resolution. However, they may become micro spikes after integration, and can then be handled by the same method at the five-minute resolution. In this way, all micro spikes within the processing window are detected and replaced by smoothed loads, whereas the load data outside the window are not touched.

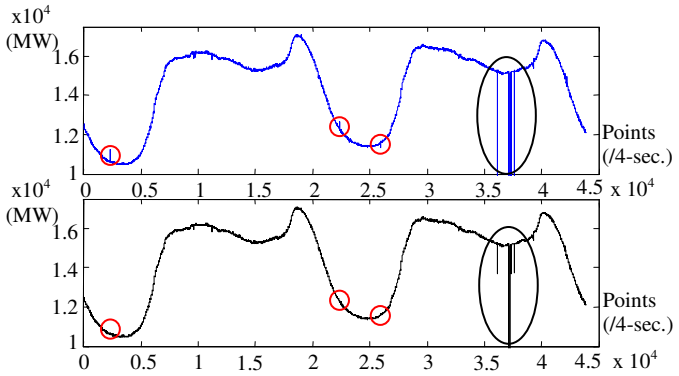

Figure 2-1. Before (top) and after (bottom) micro and macro spike filtering based on two days of continuous ISO-NE load data at the four-second resolution

2.3.2 Macro Spike Filtering

The key idea for filtering out macro spikes is to detect a pair of edges, and fix the loads between the two edges with linear interpolation values. This method is only applied to the loads at the five-minute resolution because macro spikes at the four-second resolution may become micro spikes after integration. To detect edges, the first-order differencing transformation is applied to the load series at the five-minute resolution. The edge is said to be detected when the absolute value of the difference exceeds the

threshold m. A macro spike is then said to be recognized when two sequential edges are located, and the width of the two edges is less than a threshold w_2 and equal to or greater than the threshold w_1. The spike whose width is less than w_1 is a micro spike, and should have been removed in micro spike filtering which is described in Subsection 2.2.1. To fix a macro spike, the load in-between the two edges is replaced by a value from linear interpolation. This interpolation method is used because the changes of five-minute loads over a short time period have similar slopes for most of the times based on observation. Figure 2-2 depicts the five-minute load series (four-second integrated into five-minute loads depicted in the second plot of Figure 2-1) before and after the macro filtering is applied.

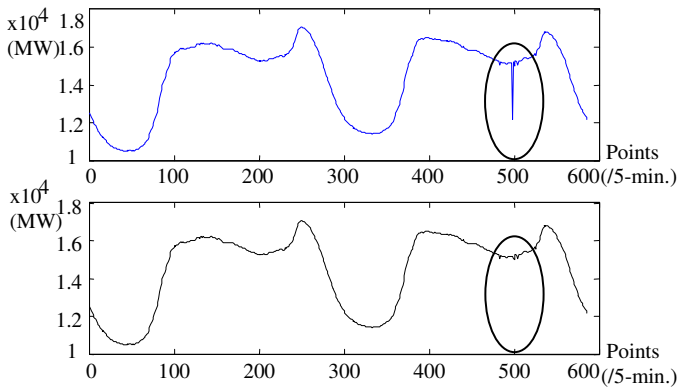

Figure 2-2. Before (top) and after (bottom) macro spike filtering at the five-minute resolution

2.4 Wavelet Neural Networks

To perform accurate predictions after pre-filtering, load properties are analyzed in Subsection 2.4.1. Data analysis shows that the load data have different components: a

very fast changing component from five to fifteen-minute resolutions, a fast changing component from fifteen-minute to one-hour resolutions, and a slow changing component with hourly, weekly, and monthly patterns. The WNN method is developed to capture the complicated load properties. To accurately capture load features at multiple frequencies, a wavelet technique is used to decompose the loads into several frequency components in Subsection 2.4.2. Due to the use of convolution in the wavelet transform, additional data need to be padded at the end side of the load segment in real-time. Relationships among the padding parameters are discussed and derived. Different padding strategies are then tested, and the best one is determined via the test data set. In Subsection 2.4.3, each load component is properly transformed and then fed with other time and date indices to a separate neural network. Predictions from individual neural networks are combined to form the forecasts. Finally, twelve dedicated wavelet neural networks are used to perform moving forecasts in Subsection 2.4.4.

2.4.1 Load Property Analysis

Very short-term load data have complicated properties. They are illustrated by the power spectrum density which describes how the power of load data is distributed with frequency. As shown in Figure 2-3, the main power lies in the low frequency and several small pedals afterward, and each one has a unique frequency component. Intuitively, this frequency domain is divided into three frequency components as denoted by low, medium, and high frequencies. If each one is further magnified by amplitude spectrum (explained in the next paragraph), it is observed that these components have different features.

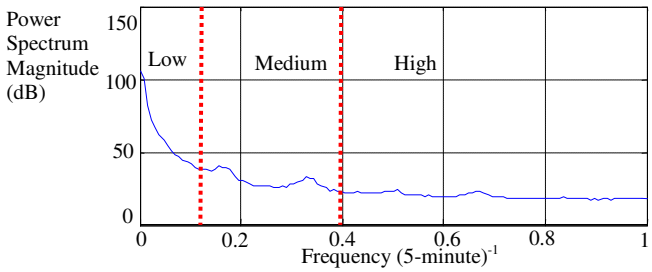

Figure 2-3. Power spectrum density for five-minute load data (January 1st, 2007 to June 30th, 2008)

As depicted in Figure 2-4, the amplitude spectrum shows that the low, medium, and high load frequency components have different features. Spectral lines for the low frequency component in the eclipse are magnified further. These spectral lines represent unique load patterns, and the ones located at frequencies corresponding to hourly, weekly, and monthly information are marked. The amplitude spectrums for the medium and high load frequencies (reflecting fast changes in load data) have small magnitudes, and hence are not magnified. Dashed lines are used to separate load components as they are in the separation in Figure 2-3.

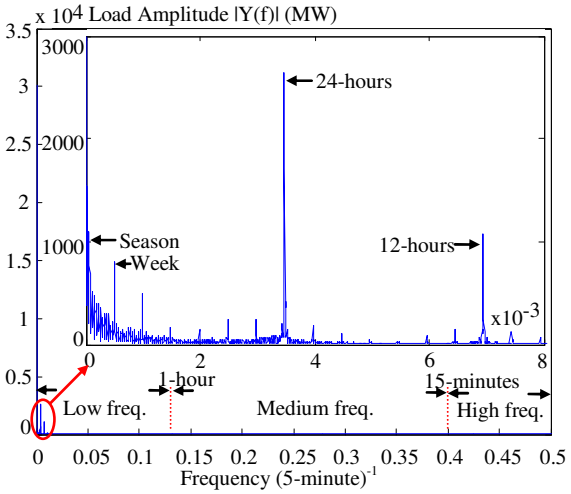

Figure 2-4. Amplitude spectrum for five-minute load data

2.4.2 Filter Bank in Wavelet Transform

The load data have multiple frequency components as depicted in Figure 2-3, and each may have a unique pattern as depicted in Figure 2-4. An intuitive idea is to decompose the loads into multiple frequency components and process each independently. For example, the load data were decomposed into multiple resolution scales in (Rocha Reis and Alves da Silva, 2005; Benaouda et al., 2006). Fourier transform is a straightforward technique to represent the signal as a sum of sinusoids which are only localized in frequency. In contrast to Fourier transform, wavelets are localized in both time and frequency and often give a better representation using multi-resolution analysis. A detailed introduction to wavelets can be found in (Mallat, 2009: Chapter 1). Motivated by the successful one-level wavelet decomposition for short-term load forecasting in our previous work (Chen et al., 2010), a wavelet technique is chosen to decompose input loads into multiple frequency components. The input loads are first decomposed into low (L) and high (H) frequency components at level one. The L frequency component called "approximation" represents a general trend of the signal, whereas the H frequency component is viewed as a difference between two successive approximations (Rocha Reis and Alves da Silva, 2005). Since the load has a large magnitude and multiple frequency information, the L frequency component is further decomposed into low-low (LL) and low-high (LH) frequency components. There is no need to decompose the H component because it has a small magnitude as compared to the L frequency component. The decomposed level is analyzed later on. Components LL,

LH, and H are very similar to the low, medium, and high frequencies described in Subsection 2.4.1.

To implement the two-level wavelet transform, a three-channel filter bank is used as shown in Figure 2-5. The high frequency channel consists of the analysis and synthesis stages. At the analysis stage, a high pass filter (a wavelet function that plays the role of the anti-alising) G_1 filters out the low frequency component. A down-sampling step then removes the odd-numbered data points. At the synthesis stage, the up-sampling step pads zeros to down-sampled data to recover the data length. A high pass filter H_1 then removes the replicas of signal spectrum caused by up-sampling. Similarly, the low-high frequency channel uses a low pass filter G_0 to compute the general trend, and then holds the even-numbered points. Next, these points are further decomposed into two parts. The low-high part convolves with G_1 and then takes steps similar to those for the high frequency channel. To recover the initial input length, the output from H1 has to be up-sampled and convolve with H_0. These are the steps to produce the LH frequency component. The same is true for the low-low frequency channel. Filters G_0, G_1, H_0, and H_1 have to satisfy perfect reconstruction and orthogonality in (Strang and Nguyen, 1997).

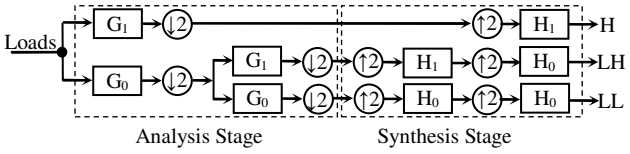

Figure 2-5. Three-channel filter bank

The filter bank in the wavelet transform described above adopts a circular convolution as explained in (Strang and Nguyen, 1997: Chapter 8). Circular convolution causes boundary distortions which affect neural network predictions. To reduce the distortion, it is necessary to extend the signal beyond the boundaries. In the high frequency channel shown in Figure 2-5, the distortion length for a convolution between the input loads and G_1 is (lw-1) based on the convolution theory, where lw is the filter length. Down-sampling and up-sampling do not produce the distortion. H_1 introduces another distortion with the same length (lw-1). The total distortion length is 2(lw-1). The low-high frequency channel sequentially convolves the inputs with four filters (G_0, G_1, H_1, and H_0) with a final length of the distortion 4(lw-1), doubling that of the high frequency channel. The same is true for the low-low frequency channel. The distortion length is thus roughly doubled for a component which is further decomposed one more level. A detailed analysis can be found in (Guan and Luh, 2010). To make sure that at least one value is not affected by distortion, the load inputs to NN need to be padded. The padding length has to be equal to or greater than the distortion length (wmaxlev function in MATLAB Wavelet Toolbox):

$$lx = (lw - 1) \cdot 2^{lvl}, \tag{3}$$

where lx is the distortion length which indicates the minimum padding length, and lvl is the level of the decomposition. Hence, the total length for load inputs to be decomposed has to be equal to or greater than the sum of the minimum padding length lx and the length of the load inputs of the last hour (12 points). For VSTLF, the latest historical data are available and used to pad the last hour's loads at the front. Additional data are needed to pad the last hour's loads at the end, as discussed in the end of this subsection.

From (3), the relationships among the decomposition level *lvl*, the filter (G and H) length *lw*, and the minimum padding length lx are very close. It can be concluded that fixing *lvl* and increasing *lw*, or vice versa, will increase *lx*. This indicates that the padding length will increase, which may not be good because a long padding to load inputs can result in a poor training and prediction for NN. However, *lvl* should not be too small because the features of load components cannot be fully captured. The same is true for *lw* because a small *lw* has a poor ability to represent the load component behaviors. It is clear that neither *lw* nor *lvl* should be too large or small, so that a reasonable lx can be obtained. Therefore, a balance among *lvl*, *lw*, and *lx* has to be made due to their close relationships in (3).

To choose a good *lvl* for decomposition, different values are tested and compared, while *lw* and *lx* remain fixed. Two-level decomposition is found to be the best among levels from zero to three in Example 2 in Section 2.5. This corresponds to the scheme presented in Figure 2-5 with three decomposed frequency components H, LH, and LL. To choose a good filter length *lw*, a proper wavelet has to be chosen using the previous fixed lx and newly determined *lvl* (=2). Daubechies (Db) wavelets are adopted in our method because they belong to a family of orthogonal wavelets and are characterized by frequency responses having maximum flatness (at 0 and π). Db members tested are Db2-Db20 (even index numbers only). The index number refers to the filter length lw, and has the ability to represent complicated behaviors of signal components. For example, Db2 encodes constant components, and Db4 encodes constant and linear ones. However, the Db number cannot be too large. Otherwise, the minimum padding length will increase. Based on observation, the changes of load series over a short time period have

similar slopes for most of the times. Hence, Db4 seems to be a reasonable choice because it encodes linear signal components, and is demonstrated to be the best among all the index numbers tested as presented in Section 2.5.

Once *lvl* (=2) and *lw* (=4) are fixed in equation (3), lx can be calculated ((*lw*-1)·2^{lvl}=12). Since the last hour's loads (12 points) are used as NN inputs, the total length for load inputs to be decomposed has to be equal to or greater than the sum of the minimum padding length and the length of the last hour's loads (i.e., the total length ≥24). A more precise number can be calculated from the derivation in (Guan and Luh, 2010). To further reduce distortion effects, padding strategies (e.g., zero-padding, periodic extension, and symmetrization) are tested. According to the test in Example 2 in Section 2.5, symmetrization, a boundary replication which pads the loads by adding points symmetric to the original, is demonstrated to be the best strategy. This also corresponds with the conclusion on page 263 in (Strang and Nguyen, 1997). These parameters are determined through training, validation, and test processes in a three way data split.

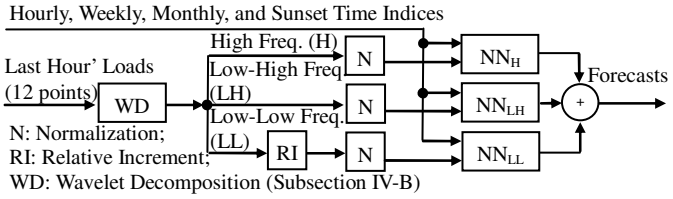

Figure 2-6. Structure of wavelet neural networks

2.4.3 Neural Networks

To capture decomposed frequencies, our idea is to properly transform individual components as presented in Subsection 2.4.1. The transformed components are then fed

to separate neural networks. Finally, individual predictions from NNs are added to form the forecasts as depicted in Figure 2-6.

The load components are treated differently. The LL frequency represents the majority of load information, including hourly, weekly, and monthly patterns as analyzed in Subsection 2.4.1. Since the loads from 5 to 60 minute outs are predicted each time, the loads of the last hour (lag=12) are used as inputs. Loads with other lags are also tested, but the results are not further improved. To remove a first-order trend and anchor the predictions by the latest load, the relative increment (RI) in loads in (Charytoniuk and Chen, 2000), is applied:

$$LL_d^{RI}(t) = \left(LL_d(t) - LL_d(t-1)\right) / LL_d(t-1), \qquad (4)$$

where LL represents the low-low frequency load component at day index d, and t is the time index in a five-minute period. RI indicates the relative increment transformation and is used to stationarize the load component series. This transformation reveals more of the hidden information in the LL frequency component in Figure 2-7.b than the one without applying RI in Figure 2-7.a. But the other observation shows that RI reveals less of hidden information in the LH and H components than the one without applying RI. Hence, RI transformation is only applied to the LL load component. All the components then have to be normalized and fed to individual NNs.

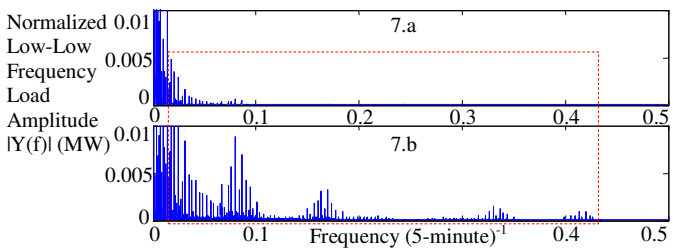

Figure 2-7.a. Amplitude spectrum for normalized low-low frequency before applying RI; Figure 2-7.b. Amplitude spectrum for normalized low-low frequency after applying RI

In addition to load inputs (5 to 60 minutes), time and date indices are parts of the neural network inputs, including hourly, weekly, and monthly indices. Furthermore, sunset time is included to capture the load feature related to the street lighting. These indices are used to help NNs indentify the periodical patterns of load data. Similarly, low-high and high frequency NNs adopt the same time and date indices but use the load components without RI transformation. Finally, results from three NNs are summed up to form final forecasts. Other additional inputs were tested but not considered because the results were not significantly improved. These inputs include: area control errors, frequencies, and some selected loads from history (e.g., loads of the last several hours, loads of selected hours from yesterday, similar day's loads, and so on). Based on the literature review, actual weather data and weather forecasts from related methods, e.g., the climatology method (weather world 2010 project), are seldom used for VSTLF inputs because of the large time constant of the load and weather relationship. Also, real-time weather data are not available from ISO New England.

To narrow the numerous choices of input candidates down, different combinations of data inputs are screened based on small data sets. For example, load data from November 2007 to December 2007 are used for training, and loads for January 2008 are then predicted. The resulting candidate inputs are then examined through training, validation, and test processes in a three way data split.

2.4.4 Moving Forecasts

When performing moving forecasts every five minutes, the intuitive approach would be to train a single WNN offline with historical data as presented in (Haykin, 2009: Chapter 4) and train the WNN online whenever a new data point is available. This is the same as the self-adaptive training process of (Herrera et al., 2010). However, test results using this approach are not satisfactory. Based on further testing, our final configuration consists of twelve dedicated WNNs, one for each five-minute period in the hour. In this way, individual WNNs can be properly trained. For example, at 2:55 am, WNN_1 predicts the loads from 3:00 am to 3:55 am in five-minute periods; WNN_2 at time 3:00 am predicts the loads from 3:05 am to 4:00 am in five-minute periods, etc. Then at 3:55 am, when the measured values are known for the past 12 time steps, WNN_1 is trained online (updated) with the data from 3:00 am to 3:55 am and then predicts the loads from 4:00 am to 4:55 am, and the process repeats.

2.5 Numerical Test Results

The method was developed in MATLAB for prototype implementation and then converted to JAVA using Eclipse. The open source can be downloaded at http://github.com/ldmbouge/vstlf. In this section, the software was run on a server with dual Xeon quad core Intel E5620 2.4GHz processors and 36 GB memory. The performance measures include mean absolute error (MAE), mean absolute scaled error (MASE) as presented in (Hyndman and Koehler, 2006; Hyndman, 2006), mean average percentage error (MAPE), and standard deviation of sample errors (SD):

$$MAE(k) = n^{-1} \sum_{t=k}^{12n+k} \left| L_p(t) - L_A(t) \right|, \quad k = 1, \cdots, 12 , \qquad (5)$$

$$MASE(k) = MAE(k)_{out-of-sample} / MAE_{in-sample}\ , \qquad (6)$$

$$MAPE(k) = n^{-1}\sum_{t=k}^{12n+k}\left(\left|L_p(t) - L_A(t)\right| / L_A(t)\right)\times 100\%\ , \qquad (7)$$

$$SD(k) = \left[n^{-1}\sum_{t=k}^{12n+k}\left(L_p(t) - n^{-1}\sum_{t=k}^{12n+k}\left(L_p(t) - L_A(t)\right)\right)\right]^{\frac{1}{2}}. \qquad (8)$$

In the above equations, index k represents 5 to 60 minutes in five-minute steps, n indicates the number of hours in the forecasting horizon, and $L_A(t)$ and $L_P(t)$ denote actual and predicted loads at sample time t, respectively. The general performance measures include MAE, MAPE, and SD. MASE provides a scale-free error metric for comparing forecasting methods on a single series (Hyndman and Koehler, 2006). In (6), the numerator $MAE(k)_{out\text{-}of\text{-}sample}$ for k-step out (k = 1, …, 12) is calculated for the multistep WNN forecasts computed out-of-sample (in the testing data set). The denominator $MAE_{in\text{-}sample}$ is calculated for the one-step "naïve forecast" computed in-sample (in the training and validation data sets). The naïve forecast for each future period is the actual value for the previous period (Hyndman, 2006). This denominator $MAE_{in\text{-}sample}$ is used to scale the numerator $MAE(k)_{out\text{-}of\text{-}sample}$ to generate a scale-free error metric that is stable, easy to compute, and in the correct unit. If the MASE value is less than one, this indicates that the forecast of the presented method is better than the one-step naive forecast. However, if the MASE value is greater than one, this indicates the opposite. Multistep MASE values are often larger than one as the forecasting horizon increases because one step naive forecast is used for scaling (Hyndman and Koehler, 2006; Hyndman, 2006). Equations (5) to (8) can also be applied to moving forecasts with multiple WNNs.

Two examples are presented to demonstrate our method. Example 1 uses a classroom-type problem to compare a single (standard) NN to our two-level wavelet NNs

so that our method can be duplicated and verified in a simple way. Example 2 demonstrates the values of spike filtering methods, two-level decomposition, Db4 wavelet, Symmetrization padding, selected time and date indices (hourly, weekly, monthly, and sunset time), and relative increment transformation to the LL frequency component.

In both examples, standard neural networks based on the back-propagation learning algorithm in (Haykin, 2009: Chapter 4) are used. The training, validation, and test processes in a three way data split are used to determine and demonstrate the parameters in the model. All NNs are trained offline by using historical data with weights randomly initialized, and the training terminates when the stopping criteria is reached to be described in Examples 1 and 2. These NNs are then trained online with the latest twelve loads as explained in Subsection 2.4.3

Example 1. Consider the signal:

$$y(t) = 100\sin(2\pi 10t) + 20\sin(2\pi 150t) + \sin(2\pi 200t), \quad (9)$$

where the signal y(t) is composed of a low frequency component $100\sin(2\pi 10t)$, a medium component $20\sin(2\pi 150t)$, and a high component $\sin(2\pi 200t)$. The signal is similar to the actual load in terms of relative amplitude and frequency. A total of 3600 data points $(t, \tilde{y}(t))$ were randomly generated:

$$\tilde{y}(t) = y(t) + \varepsilon(t), \quad (10)$$

where $t \in [1, \ldots, 3600]$ and $\{\varepsilon(t)\}$ were independent and identically distributed normal noises with zero mean and unit variance N(0, 1). The first one-third of data points were

used for training, the second one-third of data for validation, and the last one-third of data for testing.

A single NN without wavelet decomposition is compared to neural networks with two-level wavelet decomposition. The relative increment transformation is not used for this example because $y(t)$ consists of three sine functions which are periodical, and there is no need to use this transformation to make $\{y(t)\}$ stationary. Based on the training, validation, and test processes in a three way data split, the number of hidden neurons for the standard NN method is set to be 11, and the numbers of hidden neurons for our method are set to be 8, 7, and 13 for H, LH, and LL NNs, respectively. For both methods, NN training processes stop when MAE thresholds (stopping criteria) are reached. From the test data set, the overall MAE and SD are respectively 1.73 and 2.33 for standard NN method, whereas the overall MAE and SD are respectively 0.85 and 1.06 for our method. MAPE is not adopted since $\{y(t)\}$ may have zero values. MAEs and SDs indicate that the predictions obtained from using two-level wavelet NNs are both closer to the true values in data series $\{y(t)\}$ and have smaller standard deviations than the ones obtained using a single NN.

Example 2. Wavelet neural networks with spike filtering are tested with system load data provided by ISO New England. The training period is from January 1st, 2007 to December 31th, 2007, the validation period is from January 1st, 2008 to June 30th, 2008, and the test period is from July 1st, 2008 to December 31th, 2009. Ten cases are tested. Since there are many factors in setting the forecasting model, and each factor has multiple options, the number of possible combinations of options is very large. To have a

practical way to demonstrate the appropriateness of options selected for individual factors, the configuration determined through training, validation, and test processes is treated as the nominal configuration. Based on it, each factor is then examined in individual cases below. Cases 1 -7 are for training and validation: Case 1 for micro and macro spike filtering, Case 2 for spike filtering thresholds, Case 3 for decomposition levels, Case 4 for selecting Daubechies wavelets, Case 5 for padding strategies, Case 6 for date and time indices, and Case 7 for relative increment transformation. Cases 8-10 are for testing: Case 8 for test results and prediction interval construction, Case 9 for comparing with ISO-NE's method, and Case 10 for Monte Carlo simulations.

To reduce computation time, Cases 3-7 are based on WNN_1 because its results are very similar to the individual results from other WNNs as reported in Table 2-1, while the other cases are based on the twelve dedicated WNNs. For all the cases, there are three layers in all the neural networks: one input layer, one hidden layer, and one output layer. Through training, validation, and test processes in a three way data split, the numbers of hidden neurons are 6, 13, and 18 for H, LH, and LL NNs, respectively. They are not identical because the decomposed load components have different features. Based on testing, a single WNN is trained offline for three hours (stopping criterion), and twelve WNNs require a total of thirty six hours for training offline.

Table 2-1. MAEs (MW) FOR MULTIPLE WNNS

Min.	WNN_1	WNN_2	WNN_4	WNN_6	WNN_8	WNN_{10}	WNN_{12}
5	13.43	13.50	13.49	13.45	13.46	13.42	13.48
10	19.90	19.97	19.95	19.97	19.94	19.90	19.99
15	25.60	25.75	25.75	25.73	25.71	25.59	25.61
20	31.56	31.68	31.79	31.73	31.72	31.55	31.57
25	36.88	36.99	37.07	36.99	37.07	36.87	36.81

30	42.48	42.65	42.75	42.66	42.70	42.46	42.40
35	48.09	48.29	48.43	48.31	48.31	48.06	47.99
40	53.83	54.09	54.18	54.05	54.05	53.79	53.66
45	59.23	59.55	59.59	59.49	59.56	59.22	59.01
50	64.74	65.18	65.21	65.16	65.21	64.73	64.55
55	69.26	69.71	69.63	69.69	69.69	69.24	69.11
60	74.40	74.86	74.78	74.97	74.89	74.38	74.27

CASE 1. Spike filtering methods are tested with ISO-NE's real-time load data. Results for multiple WNNs with the loads filtered by the micro and macro filters are compared to the ones with unfiltered loads, the loads only filtered by the micro filter in four seconds, and the loads only filtered by the macro filter. The results for 5 to 60 minute outs in Table 2-2 show that both micro filtering in four seconds and macro filtering improve MAEs and SDs. Furthermore, using the micro and macro spike filtering together produces the smallest MAEs and SDs, and these results are treated as nominal ones and will be used later in Cases 8 and 10 for comparisons.

Table 2-2. MAPEs (%), MAEs (MW), AND SDs (MW) FOR MULTIPLE WNNs IN MOVING FORECASTS WITH AND WITHOUT SPIKE FILTERING METHODS

Min.	With loads not filtered			With loads only filtered by the micro filter in four seconds		
	MAPE	MAE	SD	MAPE	MAE	SD
5	0.10	15.56	17.77	0.09	13.56	16.24
10	0.15	22.78	27.09	0.13	20.08	25.04
15	0.23	35.56	41.94	0.17	25.95	32.47
20	0.31	47.69	56.36	0.21	32.00	39.57
25	0.38	57.06	68.12	0.25	37.46	46.47
30	0.44	67.32	80.83	0.29	43.27	53.54
35	0.53	80.05	95.49	0.32	49.09	60.40
40	0.61	92.63	110.24	0.36	55.01	67.24
45	0.68	103.96	123.73	0.40	66.60	73.90
50	0.76	115.77	137.60	0.44	66.47	80.65
55	0.79	121.35	142.87	0.47	71.16	85.83
60	0.85	129.30	151.05	0.50	76.54	91.73
Min.	With loads only filtered by the macro filter			With loads filtered by the micro & macro filters		
	MAPE	MAE	SD	MAPE	MAE	SD
5	0.09	13.52	16.19	0.09	13.49	16.03
10	0.13	20.05	24.99	0.13	20.00	24.43

15	0.17	25.92	32.41	0.17	25.65	30.71
20	0.21	31.97	39.51	0.21	31.61	36.89
25	0.25	37.42	46.40	0.24	36.88	42.97
30	0.29	43.23	53.46	0.28	42.45	49.13
35	0.32	49.04	60.34	0.32	48.05	55.20
40	0.36	54.97	67.20	0.36	53.71	61.25
45	0.40	60.56	73.80	0.39	59.06	66.87
50	0.44	66.41	80.51	0.42	64.59	72.46
55	0.47	71.09	85.67	0.45	69.14	77.85
60	0.50	76.47	91.56	0.49	74.28	83.57

Case 2. To detect spikes by micro or macro filtering, three thresholds m, w_1, and w_2 should be determined. Based on observation, spike magnitudes are usually greater than 40MW for ISO-NE's load data, the widths of micro spikes are less than 3 points, and the widths of macro spikes are less than 10 points. Through testing based on a three way data split, the nominal values for m, w_1, and w_2 are set to be 50, 3, and 10, respectively. To partially validate this choice, different values of m are examined when w_1 and w_2 are fixed at their nominal values. MAEs and SDs in Table 2-3 show that the results with different m values are quite similar, and the configuration with m = 50 produces the best forecasting accuracy. The same steps are separately taken for the widths w_1 and w_2, and 3 and 10 are chosen, respectively.

Table 2-3. MAEs (MW) AND SDs (MW) FOR SPIKE FILTERING METHODS WITH DIFFERENT M VALUES

Min.	m=45		m=50		m=55		m=60	
	MAE	SD	MAE	SD	MAE	SD	MAE	SD
5	13.6	16.3	13.5	16.0	13.5	16.2	13.5	16.3
10	20.0	25.0	20.0	24.4	20.0	24.9	20.0	25.1
15	25.9	32.1	25.7	30.7	25.9	32.4	25.9	32.4
20	31.8	39.1	31.6	36.9	31.9	39.7	32.0	39.5
25	37.2	45.5	36.9	43.0	37.3	46.3	37.3	46.0

30	42.8	52.1	42.5	49.1	43.0	53.2	43.0	52.7
35	48.5	58.9	48.1	55.2	48.8	60.7	48.7	59.8
40	54.3	65.5	53.7	61.3	54.7	67.9	54.6	66.6
45	59.8	71.7	59.1	66.9	60.3	74.7	60.1	73.1
50	65.5	78.1	64.6	72.5	66.0	81.5	65.8	79.6
55	70.0	83.3	69.1	77.9	70.5	86.6	70.3	84.8
60	75.2	89.1	74.3	83.6	75.9	92.3	75.5	90.6

Case 3. Wavelet decomposition results from zero level (a single NN without wavelet decomposition) to three levels are compared. MAEs presented in Table 2-4 show that two-level wavelet neural networks produce the best forecasting accuracy.

SDs for decomposition levels one to three show insignificant differences, and hence will not be given in Cases 4-7.

Table 2-4. MAEs (MW) AND SDs (MW) FOR WNN1 WITH DIFFERENT DECOMPOSITION LEVELS

Min.	0-Level	1-Level		2-Levels		3-Levels	
	MAE	MAE	SD	MAE	SD	MAE	SD
5	15.64	17.94	31.80	13.43	15.92	19.66	55.02
10	24.83	25.54	35.46	19.90	24.40	27.16	57.66
15	32.89	30.93	38.56	25.60	30.85	30.57	58.63
20	40.24	36.47	42.44	31.56	37.19	36.46	61.16
25	47.64	41.91	46.65	36.88	43.29	41.90	63.62
30	54.93	47.18	51.42	42.48	49.47	47.55	67.42
35	62.96	53.44	56.40	48.09	55.42	52.48	71.19
40	70.78	59.58	61.62	53.83	61.38	58.14	75.00
45	78.52	65.25	67.63	59.23	67.08	63.42	79.94
50	86.41	71.26	73.26	64.74	72.75	69.38	85.17
55	94.06	78.20	79.49	69.26	78.12	74.06	89.05
60	102.04	84.38	85.44	74.40	83.85	79.11	93.59

Case 4. Based on the two-level wavelet decomposition, results using different Daubechies wavelets (Db2-Db20) are compared and are partially reported in Table 2-5. MAEs indicate that the Db4 gives the best prediction accuracy. This is consistent with the analysis in Subsection 2.4.2.

Table 2-5. MAEs (MW) FOR WNN1 WITH DIFFERENT DAUBECHIES WAVELETS

Min.	Db2	Db4	Db6	Db8	Db12	Db20
5	27.43	13.43	16.83	17.93	17.15	17.65
10	29.88	19.90	25.51	25.66	25.57	26.25
15	35.71	25.60	32.40	33.43	33.28	34.33
20	39.59	31.56	39.09	39.70	39.97	41.05
25	52.42	36.88	45.20	46.33	46.59	47.62
30	56.03	42.48	51.40	52.89	52.90	53.63
35	61.08	48.09	57.74	59.62	59.52	60.41
40	66.42	53.83	64.80	67.40	66.74	67.90
45	80.91	59.23	72.19	74.89	73.87	75.40
50	85.71	64.74	79.68	83.02	81.34	83.58
55	89.69	69.26	87.29	89.08	88.31	91.25
60	95.39	74.40	94.93	93.87	94.27	97.23

Case 5. To handle distortions, different padding strategies are used, including zero padding, periodic padding with order one, and symmetrization padding. In Table 2-6, results using different padding strategies are compared, and MAEs show that the symmetrization strategy gives the best prediction accuracy.

Table 2-6. MAEs (MW) FOR THE WNN1 WITH DIFFERENT PADDING STRATEGIES

Min.	Zero	Periodic	Symmetrization
5	7250.92	668.44	13.43

10	7251.40	670.97	19.90
15	7251.77	675.08	25.60
20	7252.18	678.99	31.56
25	7252.63	682.63	36.88
30	7253.07	686.52	42.48
35	7470.42	689.50	48.09
40	7263.67	692.31	53.83
45	7132.73	690.89	59.23
50	6981.50	675.37	64.74
55	6308.64	789.13	69.26
60	7053.08	1481.67	74.40

Case 6. Beyond load inputs to NNs, the selections of time and date indices are investigated and reported in Table 2-7. The combination which includes the loads of the last hour (LD), the hourly index (HI), the weekly index (WI), the monthly index (MI), and the sunset time index (SI) gives the smallest MAEs when compared to other combinations.

Table 2-7. MAEs (MW) FOR WNN1 WITH DIFFERENT TIME INDICES

Min.	LD	LD+HI	LD+HI +WI	LD+HI +WI+MI	LD+HI+WI +MI+SI
5	35.18	29.57	21.39	19.34	13.43
10	56.64	46.10	32.46	29.08	19.90
15	79.20	59.14	42.94	38.03	25.60
20	103.78	72.01	52.56	46.06	31.56
25	127.42	86.01	63.02	53.93	36.88
30	151.52	100.10	73.84	62.08	42.48
35	183.59	116.29	84.84	70.40	48.09
40	220.04	136.21	100.10	83.61	53.83
45	247.66	150.34	107.99	88.09	59.23
50	285.25	172.55	125.81	102.34	64.74

55	301.70	177.87	129.31	103.09	69.26
60	328.09	191.38	140.44	111.18	74.40

Case 7. Components with and without relative increment transformation applied are tested. The strategy using an LL frequency component with RI, and LH and H frequency components without RI produces the smallest MAEs (e.g., 75MW for the 60-minute out) when compared to the other strategies of using LL without RI, and LH and H with RI (e.g., 300MW for the 60-minute out), and using LL, LH, and H with and without RI (very large).

Case 8. Cases 1-7 are for training and validation, and Cases 8-10 are for testing. As shown in Table 2-8, the small MAPEs, MAEs, and SDs are close to the nominal results (in the block with the loads filtered by the micro and macro filters) in Table 2-2, indicating that the parameters are properly selected. Also, the MASEs for 5 to 40 minute outs are less than one, indicating that our multistep forecasts are better than the one-step naive forecast. The MASEs for 45 to 60 minute outs are slightly greater than one. This corresponds to the explanation in the beginning of Section 2.5 that multistep MASE values will often be larger than one as the forecasting horizon increases.

Table 2-8. MASEs, MAPEs (%), MAEs (MW), AND SDs (MW) FOR OUR METHOD

Min.	MAPE	MAE	SD	MASE
5	0.09	12.52	14.61	0.23
10	0.13	18.45	19.87	0.35
15	0.16	23.72	24.67	0.45
20	0.20	29.36	30.15	0.56

25	0.24	34.49	35.48	0.65
30	0.27	39.89	41.15	0.76
35	0.31	45.36	46.48	0.86
40	0.33	51.06	52.13	0.97
45	0.35	56.32	57.52	1.07
50	0.38	61.80	63.23	1.17
55	0.42	66.06	67.77	1.25
60	0.45	71.02	72.93	1.35

To evaluate the errors for 5 to 60 minute outs, box plots, as depicted in Figure 2-8, are used to graphically depict errors through five-number summaries: sample minimum, lower quartile, median, upper quartile, and sample maximum in (Stat Trek). The figure shows that forecasting errors for 5 to 60 minute outs by our method have near zero medians, and box shapes are almost symmetric. The range, especially the outlier and inter-quartile ranges, however, gradually expand as the minute out increases, due to the fact that data uncertainty increases from 5 to 60 minutes in five-minute steps.

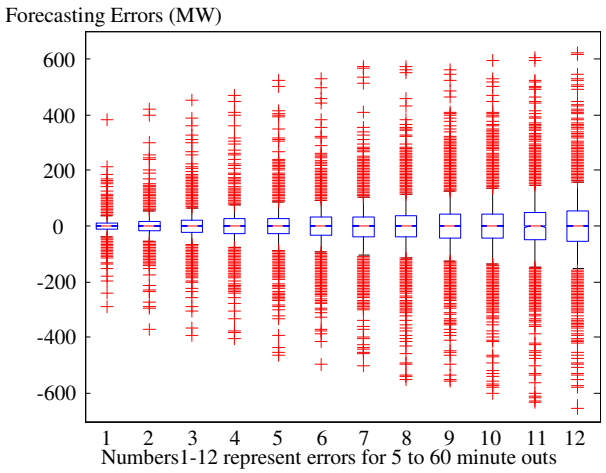

Figure 2-8. Box plots for forecasting errors for 5 to 60 minute outs

To evaluate the error bias, average errors (the mean of the differences between the actual and predicted loads) for 5 to 60 minute outs are calculated to be from -2.1 MW to 0 MW. This range is relatively insignificant compared to the overall load range, from 9000 MW to 27000 MW. This indicates that the model is almost unbiased. Furthermore, the percentages of under and over forecasts are nearly 50% for both.

To empirically construct prediction intervals, consider five-minute outs as an example. At time t, historical five-minute errors (actual minus predicted loads) before time t are ordered. For a nominal coverage rate 1-α, e.g., α=0.1, the lower and upper bounds of the 90% prediction interval are determined and then added to the forecast at time t to be the approximated prediction interval. For our testing, the errors from July 1st, 2008 to November 30th, 2008 (> 40000 errors) and the prediction are used to construct the prediction interval for t = 00:05am on December 1st, 2008. The lower and upper bounds obtained are -112.75 MW and 104.52 MW, respectively, and the predicted load is 10619 MW. When the error at 00:05 am is available, this new error and previous errors are then used together for t = 00:10 am. To quantify forecasting accuracy, this process repeats until the end of December. It turns out that 87.02% of actual load data falls within approximated prediction intervals (i.e., actual percentage coverage = 87.02%). This is close to 90%, indicating that approximated prediction intervals are reasonably accurate. The same steps are taken for 10-60 minute outs, and similar results are obtained.

Prediction intervals can also be obtained based on an estimated distribution of the variable to be forecasted by using, for example, a modified bootstrap method as presented in (Fan and Hyndman, 2012) for short-term load forecasting, or an adapted resampling

method as presented in (Pinson and Kariniotakis, 2010) for wind power generation forecasting. Since the method we used provided reasonably accurate results, these methods in (Fan and Hyndman, 2012; Pinson and Kariniotakis, 2010) are not explored.

Case 9. Results of our method and of ISO-NE's method in (Shamsollahi et al., 2001) reviewed in Section 2.2 are compared based on ISO-NE's real-time data. The forecasting period for comparison is from July 1st, 2008 to July 31th, 2008. MAPEs and MAEs in Table 2-9 show that our method produces smaller errors than ISO-NE's. This demonstrates that our method is significantly better than ISO-NE's.

Table 2-9. MAPEs (%) and MAEs (MW) Comparing Our Method' Results to ISO-NE's Results

Min.	ISO-NE's Method		Our Method	
	MAPE	MAE	MAPE	MAE
5	0.26	43.74	0.08	14.37
10	0.30	50.68	0.13	21.57
15	0.34	57.99	0.16	27.80
20	0.38	64.58	0.20	34.17
25	0.43	72.29	0.23	40.06
30	0.48	80.95	0.27	46.44
35	0.53	90.43	0.31	52.74
40	0.60	100.76	0.35	59.21
45	0.64	109.41	0.38	65.46
50	0.70	119.12	0.42	71.83
55	0.75	127.81	0.45	77.43
60	0.81	138.33	0.49	83.54

Beyond the comparison above, it is difficult to compare our results to others since there is no standard test data set for a fair comparison. Nevertheless, the following results have been reported in the literature: the MAPEs for a United States power utility (Charytoniuk and Chen, 2000) range from 0.4% to 1.1% for 20-60 minute outs in 10-minute periods; the MAPEs for British electricity demand (Taylor, 2008) range from 0.1% - 0.5% for 1-30 minute outs in 1-minute periods; the average MAPEs for 12 5-minute periods in the Ubatuba area in Brazil (de Andrade and da Silva, 2010) are 2.62%, 0.39%, and 18.72% for ARIMA, NN, and the adaptive neuro-fuzzy system, respectively; the MAPEs for a 5-minute out for the state of New South Wales in Australia (Koprinska, 2010), are 0.27%, 0.28%, 0.33%, and 0.27% for least regression, least mean square, BPNN, and support vector regression, respectively. Since data features as well as forecasting resolutions and periods are different from paper to paper, and implementation details are not open, it is difficult to evaluate individual performances. However, our method seems to be very competitive.

Case 10. To test the robustness of our method, two sets of Monte Carlo simulations are performed each with N=20 simulations. The first set of Monte Carlo simulations is run with a random weight initialization. Since N simulations are independent, the mean μ and standard deviation σ are calculated for MAPEs, MAEs, and SDs. Results in Table 2-10 show that the means μ_{MAPE}, μ_{MAE}, and μ_{SD} for 5 to 60 minute outs are close to the nominal MAPE, MAE, and SD in the loads filtered by the micro and macro filters for all the cells, as reported in Table 2-2. Also, the standard deviations σ_{MAPE}, σ_{MAE}, and σ_{SD} are small. This indicates that our method is robust.

Table 2-10. MEANS AND STANDARD DEVIATIONS FOR MAPES (%), MAES (MW), AND SDS (MW) FROM MONTE CARLO SIMULATIONS WITH A RANDOM WEIGHT INITIALIZATION (WITH N=20 SIMULATIONS)

Min.	μ_{MAPE}	σ_{MAPE}	μ_{MAE}	σ_{MAE}	μ_{SD}	σ_{SD}
5	0.09	0.00	12.81	0.64	16.42	3.42
10	0.13	0.00	19.29	2.27	21.59	2.76
15	0.16	0.00	23.93	0.45	25.94	2.30
20	0.20	0.00	29.56	0.47	31.19	1.99
25	0.24	0.01	34.70	0.59	36.42	1.83
30	0.27	0.01	40.17	0.74	41.97	1.80
35	0.31	0.01	45.65	0.88	47.32	1.98
40	0.35	0.01	51.36	1.06	53.01	2.29
45	0.38	0.01	56.68	1.24	58.42	2.65
50	0.42	0.01	62.21	1.47	64.14	3.09
55	0.45	0.01	66.51	1.54	68.67	3.16
60	0.48	0.01	71.51	1.67	73.82	3.36

The second set of Monte Carlo simulations is run with a random re-sampling step (Herrera et al., 2010). For example, time t is randomly selected from the test data set, and then historical data from one-year before t are used for training offline, and the loads one month after t are to be predicted. In comparison to the results using a random weight initialization, results in Table 2-11 show that the means μ_{MAPE}, μ_{MAE}, and μ_{SD} for 5 to 60 minute outs are close to the ones in Table 2-10. The standard deviations σ_{MAPE}, σ_{MAE}, and σ_{SD} are slightly larger than the ones in Table 2-10 for most of the cells, due to the complicated load features. However, the standard deviations for the random re-sampling step are still small. This indicates that our method is robust, and data sets are not sparse.

Table 2-11. MEANS AND STANDARD DEVIATIONS FOR MAPES (%), MAES (MW), AND SDS (MW) FROM MONTE CARLO SIMULATIONS WITH RANDOM RE-SAMPLING STEPS (WITH N=20 SIMULATIONS)

Min.	μ_{MAPE}	σ_{MAPE}	μ_{MAE}	σ_{MAE}	μ_{SD}	σ_{SD}
5	0.09	0.00	12.43	0.81	13.58	4.38
10	0.13	0.01	18.43	1.34	19.28	3.64
15	0.17	0.01	23.75	2.01	24.60	3.32
20	0.21	0.01	29.52	2.82	30.49	3.63
25	0.24	0.02	34.76	3.54	35.94	4.10
30	0.28	0.02	40.35	4.31	41.80	4.83
35	0.32	0.03	46.20	5.44	47.45	5.92
40	0.36	0.03	52.22	6.55	53.83	7.13
45	0.40	0.04	57.74	7.72	59.26	8.35
50	0.44	0.05	63.40	8.74	65.34	9.70
55	0.47	0.05	67.74	9.29	69.72	10.18
60	0.50	0.05	72.90	10.04	75.15	10.89

2.6 Conclusion

This chapter presents a method of wavelet neural networks with data pre-filtering to forecast very short-term loads one hour into the future in five-minute steps in a moving window manner. The spike filtering methods remove spikes in real-time. This WNN method can capture the load components at different frequencies. Daubechies-4 with two-level decomposition is the best configuration, which balances the decomposed level, the filter length, and the minimum padding length for decomposition. Symmetrization is shown to be the best strategy to handle the distortion. Applying the relative increment transformation to load series enhances the load stationarity. Based on test results, twelve dedicated wavelet neural networks are used to perform moving forecasts every five

minutes. Numerical testing shows accurate predictions with small standard deviations for VSTLF based on the data set from ISO New England.

3. Hybrid Kalman Filters for Very Short-term Load Forecasting and Prediction Interval Estimation

3.1 Introduction

Very short-term load forecasting predicts the loads in electric power system one or several hours into the future in steps of a few minutes (e.g., 5-min) in a moving window manner. To quantify forecasting accuracy in real-time, the forecasting process should also estimate prediction intervals (PI) online. Accurate VSTLF with good PIs is important for resource dispatch and area generation control, and helps power market participants make prudent decisions. Based on data analysis, load series have multiple frequency components, and each may have its unique pattern, such as monthly, weekly, and hourly patterns. Effective VSTLF, however, is difficult in view of different characteristics of load components and the accurate derivation for online PI estimates.

Methods for VSTLF have been reviewed in our recent paper (Guan et al., 2013), including persistence (Fox et al., 2007), extrapolation (Wang et al., 1996; Luo and He, 2007; Zhou et al., 2005; Yang et al., 2005), time series (Liu et al., 1996; Lu et al., 2005; de Andrade and da Silva, 2010; Setiawan et al., 2009; Taylor, 2008), Kalman filters (Trudnowski, 2001; Xie et al., 1996), fuzzy logic (Liu et al., 1996; Yang et al., 2006; Kawauchi et al., 2004; de Andrade and da Silva, 2010), and neural networks (NN) (Liu et al., 1996; Shamsollahi et al., 2001; Charytoniuk and Chen, 2000). Among these methods, NNs have been widely used. A standard NN trained by back propagation was used for VSTLF in (Liu et al., 1996). To make data stationary, the load inputs to an NN were transformed by using a relative increment transformation in (Charytoniuk and Chen,

2000). A single NN, however, may not be able to accurately capture complicated load features. This is because the load series has multiple frequency components, and each may have its unique pattern. To quantify VSTLF accuracy, the PI estimates should also be produced online. Since very few of these VSTLF methods have the capability of providing online PI estimates, methods of the general prediction(s) with PI(s) will be reviewed in Subsection 3.2.1, including maximum likelihood, distribution assumptive model, resampling, Bayesian inference, and Kalman filters.

Recently, we have developed a VSTLF method using wavelet neural networks (WNN) with data pre-filtering in (Guan et al., 2013). This method will be briefly reviewed in Subsection 3.2.. The key idea was to use a wavelet technique to decompose filtered loads into three orthogonal components at different frequencies: low-low (LL), low-high (LH), and high (H) frequency components. All three NNs were applied to forecast individual components, and NNs' outputs were then combined to form forecasts. To perform the VSTLF in a moving manner, twelve dedicated WNNs were used to form the moving forecast. Since WNNs were trained by back propagation, the dynamic covariance cannot be produced for PI estimation. To quantify forecasting accuracy, a general resampling method was used for PI estimates (Guan et al., 2013). The resampling, however, may not be accurate enough to estimate PIs due to the use of the back propagation algorithm for training NNs' weights. To capture complicated load features with accurate PIs, the WNN method needs to be extended, and PIs need to be further derived.

In this chapter, our previous method of wavelet neural networks trained by back propagation (Guan et al., 2013) is further improved. By replacing the first-order back

propagation algorithm with the second-order Kalman-type algorithms, a dynamic covariance can be produced for PI estimates. A method of wavelet neural networks trained by hybrid Kalman filters (WNNHKF) is developed. It forecasts loads one hour into the future in 5-min steps in a moving window manner with associated PI estimates in real-time. The data analysis shows that the LL frequency component has a near-linear relationship between the LL load input and output measurement, whereas the LH and H frequency components have nonlinear relations. To capture the near-linear relationship between the LL input and output measurement, the extended Kalman filter is used to train a neural network (EKFNN) because the EKF is derived through linearizing a system and is good for the near-linear system. To capture highly nonlinear relationships for LH and H components, the unscented Kalman filter is used to train neural networks (UKFNN) because the UKF is good for highly nonlinear systems. Hybrid Kalman filters details will be presented in Section 3.3.

Prediction intervals for VSTLF are estimated and then evaluated in Section 3.4. To accurately estimate online PIs, the overall variance estimate is calculated by adding up three orthogonal variance estimates from H, LH, and LL frequency NNs. The estimates for H and LH components are directly obtained. The estimate for LL component is further derived because the relative increment, a nonlinear transformation, is applied to the LL component. This relative increment is used to make the LL series stationary so that the transformed series can be easily captured by the NN. To assess the PIs, the distribution of the forecasting errors is analyzed, and then PIs are thoroughly evaluated.

In Section 3.5, our model is configured by training, validation, and test processes in a three-way data split, as presented in Chapter 2 of (Ripley, 1996). Example 1 uses a

classroom-type problem to compare our WNNHKF to the methods of persistence, linear AR, single NN, and WNN so that our method can be verified in a simple way. Based on a data set from ISO New England (ISO-NE), Example 2 shows the values of EKFNN for the near-linear LL frequency component and UKFNNs for highly nonlinear LH and H frequency components. This example also demonstrates the accuracy of standard deviations derived for PI estimates. It is difficult to compare this method to others since the implementation details for other methods are not open, and there is no standard test data set. Nevertheless, it is clear that Kalman filters provide as a by-product dynamic covariance matrix for PI estimates, which, based on testing, are consistent with those calculated based on static historical errors.

A preliminary version of this paper was presented in (Guan et al., 2010) where a WNN trained by hybrid Kalman filters was established for VSTLF, and standard deviations from Kalman filters were derived for PI estimates. Based on the preliminary results, the relationships between input and output measurement for individual load components are thoroughly analyzed. The consistency of the dynamic innovation covariance to the static covariance for Kalman filters is discussed. Forecasting errors are further investigated, and PIs are then thoroughly evaluated. The results of other forecasting methods are added as the reference to be outperformed. For our method, model parameters are selected and justified based on a three-way data split, as presented in Chapter 2 of (Ripley, 1996).

3.2 Literature Review

3.2.1 Prediction Interval Estimation

Existing VSTLF methods have been reviewed in (Guan and et al., 2013). Since very few of these methods have the capability of producing the accurate PI estimate(s), methods of the general prediction(s) with PI(s) construction are reviewed in this paper. These methods mainly include the maximum likelihood method, the distribution assumptive model, the resampling method, the Bayesian approach, and Kalman-type filters.

The maximum likelihood algorithm is used to obtain a set of NN weights by minimizing an error function. As presented in (Papadopoulos et al., 2001), a traditional NN was extended with a new set of hidden neurons used for computing a variance for data noises. Based on this variance, the PI was constructed.

The distribution assumptive model assumes a certain distribution for loads or forecasting errors. A probabilistic load model in (Charytoniuk, 1999; Charytoniuk and Niebrzydowski, 1998) assumed that load data had a multivariable probability density function, and predictions with variance estimates were obtained from the conditional distribution of the load given the weather information. A normal distribution for errors was assumed in (Alves da Silva and Moulin, 2000), and the PI was constructed by multi-linear regression adapted to NNs. The method was further developed in (Chryssolouris et al., 1996) to consider effects of noisy data.

The resampling method derives the PI(s) by using subsets of available data (e.g., the load or the wind generation) or drawing sample errors randomly with replacement from a set of forecasting errors. Assuming that error samples are independent and identically distributed, the PI was estimated from a cumulative distribution function using

ordered sample errors in (Papadopoulos et al., 2001). An adapted resampling method was presented to provide prediction intervals for wind power generation in (Pinson and Kariniotakis, 2010). The method relied on a classification of recent forecast errors, a fuzzy inference model, and a multisampling resampling scheme for combining probability distributions. A modified bootstrap method was developed in (Fan and Hyndman, 2012) to estimate the distribution of short-term load forecasting. Based on this, PIs were obtained.

The Bayesian approach for an NN starts with a prior distribution of the NN's weights, and then optimized weights are determined by maximizing the posterior distribution based on historical data. Through Taylor series expansion, the prediction distribution conditioned on a new input and weights was derived and approximated as a Gaussian distribution (Wright, 1999; Zhang et al., 2003). Markov Chain Monte Carlo methods were used to calculate a covariance for PI estimate in (Wright, 1999). To improve computation efficiency, Quasi-Newton methods were applied, as presented in (Zhang et al., 2003).

Kalman-type filters have been applied to NNs with PI estimates. Standard NNs are based on back propagation, which is a first-order gradient method and cannot produce a dynamic covariance for PI estimates. Therefore, the EKF was used to train and update a feed-forward NN by treating NN's weights as a state vector (Singhal and Wu, 1989). To improve computation efficiency, the EKF was extended to the decoupled EKF by ignoring the interdependence of mutually exclusive groups of weights in (Puskorius and Feldkamp, 1991). The numerical stability and accuracy of the decoupled EKF were

further improved by U-D factorization in (Zhang and Luh, 2005) for short-term load price forecasting.

Among all the methods described above, NNs have been widely used, and they provide valuable information for PI estimate(s). However, few papers have presented effective and efficient ways to produce accurate online PIs for VSTLF.

3.2.2 Wavelet Neural Networks

Recently, we have developed a method of wavelet neural networks with spike pre-filtering for VSTLF (Guan et al., 2013). The schematic of WNN is highlighted in Figure 3-1. The key idea for WNN was to use a wavelet technique to decompose the pre-filtered load data into three orthogonal components at different frequencies: LL, LH, and H components. The relative increment transformation in (Charytoniuk and Chen, 2000) was applied to the LL component to make the series stationary. The date and time indices were used to help NNs identify the periodical patterns of load data. Separate NNs were then used to predict individual components, and results of NNs were combined to form forecasts. However, it should also estimate PIs in order to quantify the forecasting accuracy in real-time. Since the WNN trained by back propagation cannot produce a dynamic covariance for PI estimates, the WNN method needs to be further improved.

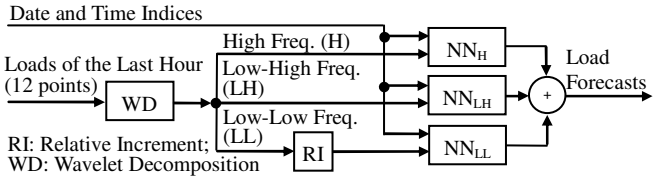

Figure 3-1. Schematic of the wavelet neural networks (WNN)

3.3 Wavelet Neural Networks Trained by Hybrid Kalman Filters

In WNN described in Subsection 3.2.1, load data have complicated features. To accurately categorize them, individual components are thoroughly analyzed. A linear autoregressive (AR) model (with a constant term added) and a standard nonlinear NN are separately used to investigate the relationship between the input load and the output measurement. Following (Guan et al., 2013), last hour's loads (12 points) are used as inputs to both models. To perform a time series of forecasts (12 points) by using AR, the input data are time-shifted. For example, data from $l(t\text{-}11)$ to l(t) are used to forecast $l(t+1)$. Next, data from $l(t\text{-}10)$ to $l(t)$ plus the prediction of $l(t+1)$ are used together to forecast $l(t+2)$, and the process repeated until a prediction is made for $l(t+12)$.

To analyze individual components, take 60-min-ahead forecasting results for example. For LL component, the coefficient of determination value is 0.97 for AR, indicating a linear mapping for LL. To explore further, the scatter plot in Figure 3-2.a shows a nonlinear pattern between the prediction (x) and the residual (y) generated by the AR model, whereas the scatter plot in Figure 3-2.d doesn't show a clear nonlinear pattern by the NN. This indicates that the AR is incapable of capturing the residual nonlinearity, while the NN is capable of capturing both linearity and nonlinearity. It can thus be concluded that the LL component has a near-linear relationship between input and output measurement. A similar analysis is conducted on the LH component. The coefficient of determination value is 0.08 for AR. Moreover, Figure 3-2.b shows predictions from AR are concentrated around zero, whereas Figure 3-2.e shows a complex pattern in predictions by the NN. The above indicates a highly nonlinear mapping for the LH

component. Similar to LH, the same conclusion is made on the H component from the coefficient of determination value as well as Figures 3-2.c and 3-2.f. Both AR and NN methods are also used for analyzing 5- to 55- min-ahead forecasting results. The result analysis again indicates that the LL component has the near-linear relationship between input and output measurement, whereas LH and H components separately have highly nonlinear relationships.

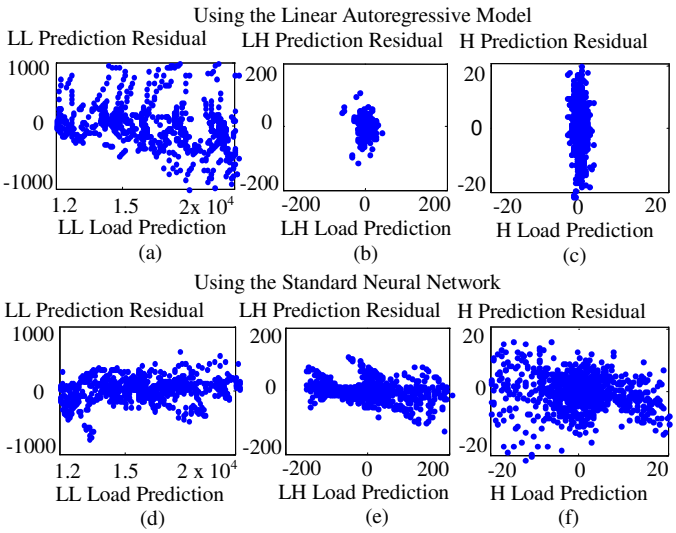

Figure 3-2. Scatter plots of 60-min-ahead predictions and residuals for individual LL, LH, and H load components (based on 1000 pair data for individual plots)

To forecast near-linear and highly nonlinear relationships for individual load components with accurate online PI estimates, the back propagation algorithm is replaced by Kalman-type filters for training WNN's weights. Generally, the back propagation is a first-order steepest decent method, whereas the Kalman filter is a second-order Newton method for recursive state estimation of linear dynamic systems, and is a minimum mean-

square-error estimator. Through treating NN's weights as a slowly varying state and the (scaled) loads as the measurement, Kalman-type algorithms are adopted because they can produce a dynamic innovation covariance whose diagonal elements can be used for PI estimates. As shown in Figure 3-3, the schematic of wavelet neural networks trained by hybrid Kalman filters is presented. To capture the near-linear relationship between the LL input and output measurement for an NN, an extended Kalman filter is used to train the neural network (EKFNN) in Subsection 3.3.1, because EKF is derived through linearizing the system and is good for near-linear systems. To capture the highly nonlinear relationships for individual LH and H components, an unscented Kalman filter is used to train the neural network (UKFNN) in Subsection 3.3.2, because UKF is good for highly nonlinear systems. Finally, results from these three NNs are added up to form forecasts. The overall variance will be derived and evaluated for PI estimates in Section 3.4.

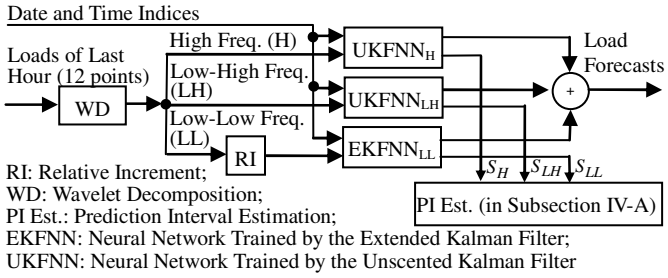

Figure 3-3. Schematic of wavelet neural networks trained by hybrid Kalman filters (WNNHKF)

3.3.1 EKFNN for the Low-Low Load Component

The key idea for forecasting the LL component is to use the EKFNN. The EKF trains the NNLL by treating its weight $w(t)$ as a slowly varying state and the (scaled) load input as the measurement $z(t)$ following (Singhal and L. Wu, 1989; Puskorius and Feldkamp, 1991; Zhang and Luh, 2005). Training an NN can be described as a state estimation problem with state and measurement equations (the symbol LL is dropped in following equations for convenience):

$$w(t+1) = w(t) + \varepsilon(t), \tag{1}$$

$$z(t) = h(u(t), w(t)) + v(t), \tag{2}$$

where $w(t)$ is an $n_w \times 1$ weight vector trained by using a set of input - output measurement pairs of an NN $\{u(t), z(t), t = 1, \ldots, T\}$, the u(t) is an $n_u \times 1$ input vector including loads of the last hour as well as the date and time indices following (Guan et al., 2013), the $z(t)$ is a corresponding $n_z \times 1$ load measurement vector (n_z is equal to 12 indicates 5- to 60- min-ahead predictions), the variable T represents a forecasting horizon, and the $h(\cdot)$ represents an input - output function of an NN. Following the standard assumption for EKF, the $n_w \times 1$ process noise $\varepsilon(t)$ is assumed to be zero-mean white Gaussian with a positive covariance $Q(t)$, and the $n_z \times 1$ measurement noise $v(t)$ is assumed to be zero-mean white Gaussian with a positive covariance $R(t)$.

In EKF, the state and covariance propagations are implemented in time-update equations. After linearizing the underlying nonlinear system, the Bayesian rule is then implemented in measurement-update equations. Following the procedure of (Bar-Shalom et al., 2001: pp. 200-210 and 382-385), key EKF steps are presented for completeness. The time-update equations are as follows:

$$\hat{w}(t+1|t) = \hat{w}(t|t), \tag{3}$$

$$P(t+1|t) = P(t|t) + Q(t), \tag{4}$$

$$\hat{z}(t+1|t) = h(\hat{w}(t+1|t), u(t)), \tag{5}$$

where the prior state (weight vector) $\hat{w}(t|t)$ and state covariance $P(t|t)$ are propagated to $\hat{w}(t+1|t)$ and $P(t+1|t)$, respectively. Here, the state transition matrix for the weight vector is an identity matrix. Next, the estimated weight $\hat{w}(t+1|t)$ together with the input $u(t)$ are used to generate the prediction $\hat{z}(t+1|t)$ which is treated as the $\hat{z}_{LL}(t+1|t)$ for the LL component. Since the function $h(\cdot)$ is nonlinear, the Taylor series expansion is used to linearize the nonlinear system, and the $H(t+1)$ is calculated:

$$H(t+1) = (\partial h(u,w)/\partial w), \ given \ \ u = u(t) \ \& \ w = \hat{w}(t+1|t). \tag{6}$$

Based on the Bayesian rule, the obtained function $H(t+1)$ is then used to produce the gain $K(t+1)$, the posterior weight $\hat{w}(t+1|t+1)$, and the state covariance $P(t+1|t+1)$. The measurement - update equations are as follows:

$$K(t+1) = P(t+1|t) \cdot H(t+1)^T \cdot S(t+1)^{-1}, \tag{7}$$

$$\hat{w}(t+1|t+1) = \hat{w}(t+1|t) + K(t+1) \cdot (z(t+1) - \hat{z}(t+1|t)), \tag{8}$$

$$P(t+1|t+1) = P(t+1|t) - K(t+1) \cdot S(t+1) \cdot K(t+1)^T, \tag{9}$$

$$S(t+1) = H(t+1) \cdot P(t+1|t) \cdot H(t+1)^T + R(t+1), \tag{10}$$

where $S(t+1)$ is an $n_z \times n_z$ innovation covariance (covariance of the measurement) and treated as the $S_{LL}(t+1)$, to be used to derive PIs in Subsection 3.4.1.

The dynamic innovation covariance S is generally consistent with the covariance calculated based on the static historical errors. This is because the state covariance P converges to a steady-state covariance under the conditions of controllability and

observability as presented on pages 211-212 of (Bar-Shalom et al., 2001). To justify these two conditions, take EKF as an example. The state transition matrix in (3) is an identity matrix, the process noise covariance Q in (4) is positive, and the measurement matrix H in (6) is believed to have a full rank given sufficient measurements. Therefore, it can be shown that the pair of state transition matrix and Cholesky factor of Q is completely controllable, and the pair of state transition matrix and H is completely observable. This yields the steady-state P and K, indicating that S is consistent with the static covariance. To demonstrate this, testing results in Example 2 of Section 3.5 show that the estimated standard deviation (derived from S) is close to the standard deviations of the sample errors. One advantage for PI estimates is that EKF can easily provide, as a by-product, an S for PI estimation. The second is that S is dynamic. Through linearizing the nonlinear system, the most recent error can be used to calculate S.

Using the EKF described above, the NN will be trained offline based on a set of input – output measurement pair data and then trained online (updated) when a new measurement is available. The EKF flowchart can be found on page 386 of (Bar-Shalom et al., 2001). For EKFNN, its load input and output are described below.

Following our previous WNN method in (Guan et al., 2013), the input LL component is transformed by using the relative increment transformation which is used to make the LL series stationary:

$$l_t^{RI} = \left(l_t - l_{t-1}\right)/l_{t-1}\,, \tag{11}$$

where l_t represents an LL load component at the time t, RI represents the relative increment transformation, and the l_t^{RI} is an element of load input vector $l_{RI}(t) = \left\{l_{t-n_z+1}^{RI},\ldots,l_t^{RI}\right\}$. To satisfy NN's input requirement, $l_{RI}(t)$ has to be normalized:

$$u_{LL}(t) = \left(l_{RI}(t) - l_{RI}^{\min}\right) \Big/ \left(l_{RI}^{\max} - l_{RI}^{\min}\right), \qquad (12)$$

where $u_{LL}(t)$ represents the normalized LL load input part at time t, and $l_{RI}^{\min}$ and $l_{RI}^{\max}$ are the minimum and maximum values of the relative increment in LL load, respectively.

After preparing NN inputs, the EKFNN performs forecasting. The forecasting output $\hat{z}_{LL}(t+1|t)$ has to be de-normalized:

$$\hat{z}^{d}(t+1|t) = \hat{z}_{LL}(t+1|t) \cdot \left(l_{RI}^{\max} - l_{RI}^{\min}\right) + l_{RI}^{\min}, \qquad (13)$$

where $\hat{z}^{d}(t+1|t)$ is a de-normalized output vector and has to be inverse-transformed with respect to the relative increment transformation in an element-wise manner. For convenience, the conditioned variable t in $\hat{z}^{d}(t+1|t)$ is dropped for all the individual elements in $\left\{\hat{z}_{t+1}^{d}, \ldots, \hat{z}_{t+n_z}^{d}\right\}$:

$$\hat{l}_{t+1} = \left\lfloor \hat{z}_{t+1}^{d} + 1 \right\rfloor \cdot l_t, \qquad (14.a)$$

$$\hat{l}_{t+2} = \left\lfloor \hat{z}_{t+2}^{d} + 1 \right\rfloor \cdot \hat{l}_{t+1} = \left\lfloor \hat{z}_{t+2}^{d} + 1 \right\rfloor \cdot \left\lfloor \hat{z}_{t+1}^{d} + 1 \right\rfloor \cdot l_t, \ldots, \qquad (14.b)$$

$$\hat{l}_{t+n_z} = \left\lfloor \hat{z}_{t+n_z}^{d} + 1 \right\rfloor \cdot \hat{l}_{t+n_z-1}, \qquad (14.c)$$

where $\hat{L}(t+1|t) = \left\{\hat{l}_{t+1}, \cdots, \hat{l}_{t+n_z}\right\}^{T}$ is the LL load prediction.

3.3.2 UKFNN for the Low-High and High Load Components

When the relationship between input and output measurement for an NN is highly nonlinear, EKF performance could be poor because the mean and covariance are propagated by linearizing an underlying nonlinear model. The key idea for forecasting LH and H frequency components is to use the UKFNN. The UKF uses an unscented transform to generate a minimal set of sample points, called sigma points, around the

mean. These sigma points are then propagated through nonlinear functions. The mean and covariance of estimates are then recovered through weighting. Because the set of sigma points are symmetrically selected, the odd central moments are zero. If the distribution for the state is multiple dimensional Gaussian, the first three moments are the same as the original moments (Julier et al., 1995). Therefore, UKF predicts the mean more accurately than EKF, and it predicts the covariance at least as accurately as EKF. It also avoids the need to calculate the Jacobian functions.

Similar to the EKF described in Subsection 3.3.1, the UKF also adopts the time-update and measurement-update equations. Rather than using the Taylor series expansion to calculate the H matrix of EKF, a set of sigma points are generated, propagated through the function, and then weighted to produce predictions with variance estimates. Following the procedure of (Julier et al., 1995), key steps of UKF are presented below for completeness. The time-update equations are the same as equations (3)-(4), where the prior state (weight vector) $\hat{w}(t|t)$ and covariance $P(t+1|t)$ are propagated to $\hat{w}(t+1|t)$ and $P(t+1|t)$, respectively. The propagations are then performed to generate a set of $2n_w+1$ sigma points χ:

$$\chi_0(t+1|t) = \hat{w}(t+1|t),$$

$$\chi_i(t+1|t) = \hat{w}(t+1|t) + \left(\sqrt{(n_w+\lambda)\cdot P(t+1|t)}\right)_i, \; i = 1,\cdots,n_w,$$

$$\chi_i(t+1|t) = \hat{w}(t+1|t) - \left(\sqrt{(n_w+\lambda)\cdot P(t+1|t)}\right)_{i-n_w}, \; i = n_w+1,\cdots,2n_w, \qquad (15)$$

where n_w is the number of NN weights, λ is a scaling parameter, and $\left(\sqrt{(n_w+\lambda)\cdot P(t+1|t)}\right)_i$ is the i^{th} column of the square root of the matrix $(n_w+\lambda)\cdot P(t+1|t)$. Through the nonlinear function $h(\cdot)$, the χ points are projected to γ points which are then weighted to produce the

NN output:

$$\gamma_i(t+1) = h(\chi_i(t+1\,|\,t), u(t)), \quad i = 0, \cdots, 2n_w, \tag{16}$$

$$\hat{z}(t+1\,|\,t) = \sum_{i=0}^{2N} W_i \cdot \gamma_i(t+1), \tag{17}$$

where W_i is the weight for the i^{th} γ point, and its definition and default value can be found in equation (15) of (Wan et al., 2000), and $\hat{z}(t+1\,|\,t)$ is the UKFNN's prediction which is treated as $\hat{z}_H(t+1\,|\,t)$ for the H component, and $\hat{z}_{LH}(t+1\,|\,t)$ for LH.

Similar to the steps for EKF, the UKFNN prediction $\hat{z}(t+1\,|\,t)$ together with χ and γ points are used to calculate the posterior weight state and covariance based on the Bayesian rule. The measurement-update equations are as follows:

$$K(t+1) = \left\{ \sum_{i=0}^{2n_w} W_i \cdot [\chi_i(t+1\,|\,t) - \hat{w}(t+1\,|\,t)] \cdot [\gamma_i(t+1) - \hat{z}(t+1\,|\,t)]^T \right\} \cdot S(t+1)^{-1}, \tag{18}$$

$$S(t+1) = \sum_{i=0}^{2n_w} W_i \cdot [\gamma_i(t+1) - \hat{z}(t+1\,|\,t)] \cdot [\gamma_i(t+1) - \hat{z}(t+1\,|\,t)]^T + R(t+1), \tag{19}$$

where the posterior weight $\hat{w}(t+1|t+1)$ and the state covariance $P(t+1|t+1)$ are as same as equations (8)-(9). The $n_z \times n_z$ innovation covariance $S(t+1)$ is treated as the $S_H(t+1)$ for the H component, and $S_{LH}(t+1)$ for the LH component. They will be used for PI estimates in Subsection 3.4.1.

Following our WNN method in (Guan et al., 2000), the H input is normalized without applying the relative increment transformation:

$$u_H(t) = \left(h_H(t) - h_H^{\min}\right) / \left(\left(h_H^{\max} - h_H^{\min}\right)\right), \tag{20}$$

where $u_H(t)$ is the normalized load component input part at time t, $h_H(t)$ represents the H load component at time t, and $h_H^{\min}$ and $h_H^{\max}$ are the minimum and maximum values of the H component series, respectively.

After input preparation, UKFNN performs the prediction which has to be de-normalized:

$$\hat{h}_H(t+1) = \hat{z}_H(t+1|t) \cdot \left(h_H^{\max} - h_H^{\min}\right) + h_H^{\min} . \qquad (21)$$

Similar to the H component, the prediction $\hat{h}_{LH}(t+1)$ can be obtained for the LH component.

3.4 Prediction Interval Estimation and Evaluation

To estimate prediction intervals online for VSTLF, the overall variance estimate is derived in Subsection 3.4.1. As shown in Figure 3-4, the key idea is to use an overall variance estimate obtained by adding up three estimates from $EKFNN_{LL}$, $UKFNN_{LH}$, and $UKFNN_{H}$. This is because these components are orthogonal based on the wavelet theory. To obtain individual variance estimates, the diagonal elements of the innovation covariance for H, LH, and LL components should be de-normalized individually. The de-normalized estimate for LL is further approximated due to the relative increment transformation. To assess the PI estimates, In Subsection 3.4.2, the Kolmogorov-Smirmov test and Quantile-Quantile plot show that the forecasting errors have heavier tails than a Gaussian distribution. Based on this, the estimated PIs are thoroughly evaluated.

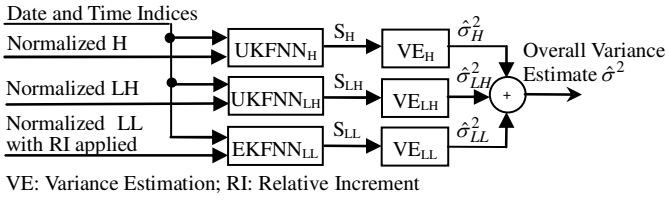

Figure 3-4. Schematic of the prediction interval estimation

3.4.1 Prediction Interval Estimation

To obtain an overall variance estimate, three variance estimates derived from individual NNs are added together:

$$\hat{\sigma}^2(t+1) = \hat{\sigma}_H^2(t+1) + \hat{\sigma}_{LH}^2(t+1) + \hat{\sigma}_{LL}^2(t+1), \qquad (22)$$

where $\hat{\sigma}^2(t+1)$ is the overall variance estimate used for online PI estimates, and $\hat{\sigma}_H^2(t+1)$, $\hat{\sigma}_{LH}^2(t+1)$, and $\hat{\sigma}_{LL}^2(t+1)$ are the individual variance estimates calculated based on $S_H(t+1)$, $S_{LH}(t+1)$, and $S_{LL}(t+1)$, respectively. To obtain the variance estimates for H and LH components, diagonal elements of $S_H(t+1)$ and $S_{LH}(t+1)$ should be de-normalized:

$$\hat{\sigma}_H^2(t+1) = \left(h_H^{\max} - h_H^{\min}\right)^2 \cdot diag\left(S_H(t+1)\right),$$

$$\hat{\sigma}_{LH}^2(t+1) = \left(h_{LH}^{\max} - h_{LH}^{\min}\right)^2 \cdot diag\left(S_{LH}(t+1)\right). \qquad (23)$$

Similarly, the diagonal ones of $S_{LL}(t+1)$ are de-normalized:

$$\hat{\sigma}_{LL}^{d\,2}(t+1) = \left(l_{RI}^{\max} - l_{RI}^{\min}\right)^2 \cdot diag\left(S_{LL}(t+1)\right), \qquad (24)$$

where $\hat{\sigma}_{LL}^{d\,2}(t+1)$ is a de-normalized variance estimate with elements $\left\{\hat{\sigma}_{t+1}^{d\,2}, \cdots, \hat{\sigma}_{t+n_z}^{d\,2}\right\}$. For convenience, the symbol LL is omitted for individual elements here as well as in the following equations. This de-normalized variance estimate then has to be further processed because the relative increment transformation is applied to the LL load input. Since the transformation is nonlinear, the derivation is difficult in view of the complicated cross-correlations for individual elements of $z^d(t+1|t)$.

The key idea for deriving the LL variance is to ignore the cross-correlations. This is because numerical testing shows that cross-correlations of the dependent elements

$\left\{z_{t+1}^d,\ldots,z_{t+n_z}^d\right\}$ in the vector $z^d(t+1|t)$ have values at 10^{-8}, whereas individual variances have values at 10^{-6}. The variance estimate is then approximated in an element-wise manner. Following 14.a, the estimate $\hat{\sigma}_{t+1}^2$ for l_{t+1} is derived:

$$\hat{\sigma}_{t+1}^2 = Var\left[\left(z_{t+1}^d+1\right)\cdot l_t\right] = \sigma_{t+1}^{d\,2}\cdot l_t^2. \tag{25}$$

Following 14.b, the $\hat{l}_{t+2}$ is calculated based on $\hat{l}_{t+1}$. By omitting their covariance, the estimate $\hat{\sigma}_{t+2}^2$ is approximated:

$$\begin{aligned}\hat{\sigma}_{t+2}^2 &= Var\left[\left(z_{t+1}^d+1\right)\cdot\left(z_{t+2}^d+1\right)\cdot l_t\right]\\ &\approx\left[Var\left(z_{t+1}^d\right)+Var\left(z_{t+2}^d\right)+Var\left(z_{t+1}^d\cdot z_{t+2}^d\right)\right]\cdot l_t^2\end{aligned} \tag{26}$$

In the equation above, the numerical testing shows that elements $z_{t+1}^{d\,2}$ and $\sigma_{t+2}^{d\,2}$ have values at 10^{-4} and 10^{-6}, respectively. Since the term $\sigma_{t+1}^{d\,2}\cdot\sigma_{t+2}^{d\,2}$ is relatively small, it is ignored. The estimate is further approximated:

$$\begin{aligned}\hat{\sigma}_{t+2}^2 &\approx\left\{\sigma_{t+1}^{d\,2}+\sigma_{t+2}^{d\,2}+E\left[\left(z_{t+1}^d\cdot z_{t+2}^d\right)^2\right]-\left[E\left(z_{t+1}^d\cdot z_{t+2}^d\right)\right]^2\right\}\cdot l_t^2\\ &=\left\{\sigma_{t+1}^{d\,2}+\sigma_{t+2}^{d\,2}+E\left[z_{t+1}^{d\,2}\right]\cdot E\left[z_{t+2}^{d\,2}\right]-\left[E\left(z_{t+1}^d\right)\cdot E\left(z_{t+2}^d\right)\right]^2\right\}\cdot l_t^2\\ &=\left\{\sigma_{t+1}^{d\,2}+\sigma_{t+2}^{d\,2}+\left(\hat{z}_{t+1}^{d\,2}+\sigma_{t+1}^{d\,2}\right)\cdot\left(\hat{z}_{t+2}^{d\,2}+\sigma_{t+2}^{d\,2}\right)-\left(\hat{z}_{t+1}^{d\,2}\cdot\hat{z}_{t+2}^{d\,2}\right)\right\}\cdot l_t^2\\ &=\left[\sigma_{t+1}^{d2}+\sigma_{t+2}^{d2}+\hat{z}_{t+1}^{d2}\cdot\sigma_{t+2}^{d2}+\hat{z}_{t+2}^{d2}\cdot\sigma_{t+1}^{d2}+\sigma_{t+1}^{d2}\cdot\sigma_{t+2}^{d2}\right]\cdot l_t^2.\end{aligned} \tag{27}$$

In the second equality above, $\hat{z}_{t+1}^d = E\left[z_{t+1}^d\right]$ and $\hat{z}_{t+2}^d = E\left[z_{t+2}^d\right]$ are based on page 203 of (Bar-Shalom et al., 2001):

$$\hat{z}^d(t+1\,|\,t) = E\left[z^d(t+1)\,|\,Z^t\right], \tag{28}$$

where $\hat{z}^d(t+1\,|\,t)$ has elements $\left\{\hat{z}_{t+1}^d,\hat{z}_{t+2}^d,\ldots,\hat{z}_{t+n_z}^d\right\}$, $z^d(t+1|Z^t)$ has elements $\left\{z_{t+1}^d,z_{t+2}^d,\ldots,z_{t+n_z}^d\right\}$, and Z^t represents the past observations up to t. This is because under the Markov assumption, the predicted measurement given the immediately previous one

is conditionally independent of the other earlier measurements.

To estimate other variances, i.e., $\hat{\sigma}^2_{t+3}$, …, $\hat{\sigma}^2_{t+n_w}$, the process will be repeated until the last element is calculated. Finally, a general equation is obtained:

$$\hat{\sigma}^2_{t+J} \approx \sum_{j=1}^{J} \left\{ \left(1 + \sum_{i=1}^{J} \hat{z}^{d\,2}_{t+i} - \hat{z}^{d\,2}_{t+j} \right) \cdot \sigma^{d\,2}_{t+j} \right\} \cdot l_t^2$$

$$= \sum_{j=1}^{J} \left\{ \left(1 + \sum_{i=1}^{J} \left(\left(l_{RI}^{\max} - l_{RI}^{\min} \right) \cdot \hat{z}_{t+i} + l_{RI}^{\min} \right)^2 \quad - \left(\left(l_{RI}^{\max} - l_{RI}^{\min} \right) \cdot \hat{z}_{t+j} + l_{RI}^{\min} \right)^2 \right) \right.$$

$$\left. \cdot \left(l_{RI}^{\max} - l_{RI}^{\min} \right)^2 \quad \cdot diag\left(S_{LL}(t+1) \right)_{t+j} \right\} \cdot l_t^2, \quad J = 1, \cdots, n_z, \qquad (29)$$

where $\hat{\sigma}^2_{LL}(t+1) = \left\{ \hat{\sigma}^2_{t+1}, \cdots, \hat{\sigma}^2_{t+n_z} \right\}^T$ is an approximated variance estimate vector for LL load component.

3.4.2 Evaluation of Prediction Interval Estimates

To help evaluate PI estimates, the distribution of forecasting errors for individual 5- to 60-min outs is analyzed. The Kolmogorov-Smirmov test and Quantile-Quantile plot of the errors show that the errors have heavier tails than a Gaussian distribution. However, the Kolmogorov-Smirmov test shows that after removing the bottom and top tails of the errors (e.g., 5-min. errors that are either below the 0.7^{th} percentile or above the 99.3^{th} percentile), the remaining errors follow a zero mean Gaussian distribution. This test is performed in two ways. First, the remaining errors are standardized without centering, and the empirical distribution of the resulting values is compared with a standard Gaussian distribution. Second, the empirical distribution of the remaining errors is compared with that of simulated data sampled from a Gaussian distribution with zero mean and the same standard deviation. Numerical details and results of these two ways

of the test are given in Case 3 of Example 2 in Section 3.5, demonstrating near Gaussian distribution of the forecasting errors except for heavy tails.

Based on the above analysis, the PI estimates are then evaluated in three ways. First, the estimated standard deviations for 5- to 60-min errors are compared with the sample ones, respectively. Second, the one sigma coverage rates based on the estimated standard deviations are compared with 68%, i.e., one sigma coverage rate of the standard Gaussian distribution. Third, for each of the coverage rates 10%, 20%, ..., 90%, we calculate how many estimated standard deviations are needed to achieve the coverage rate for the errors, and then compare the result with how many standard deviations are needed to achieve the same rate for a Gaussian random variable. As shown from the numerical results in Case 3, the comparisons indicate that the PI estimates are reasonably accurate and conservative.

3.5 Numerical Testing Results

The method was implemented in MATLAB. The open source code and the part of the test data and results are open, and can be obtained from http://github.com/ldmbouge/vstlf. For this section, the software was run on a server with dual Xeon quad core Intel E5620 2.4GHz processors and a 36 GB memory. The performance measures include mean absolute error (MAE), mean average percentage error (MAPE), standard deviation of sample errors (SD), estimated standard deviation (ESD) which is the square root of the variance estimate derived in Subsection 3.4.1, and one sigma coverage.

Two examples are presented to demonstrate our method. Example 1 uses a

classroom-type problem to compare the WNNHKF to the methods of persistence, linear AR, single NN, and WNN so that our method can be verified in a simple way. Example 2 shows the values of $EKFNN_{LL}$ for capturing the near-linear relationship between the LL input and output measurement, as well as $UKFNN_{LH}$ and $UKFNN_{H}$ for capturing highly nonlinear relationships. This example also demonstrates the accuracy of the derived PI estimates. In both examples, the training, validation, and test processes in a three-way data split are used to determine the parameters in WNNHKF. All NNs (trained by Kalman filters) are trained off-line by using training data with weights randomly initialized, and the training terminates when a fixed number of iterations are reached.

Example 1: Consider the signal:

$$y(t) = 200\sin(2\pi 10t/f_s) + 10\sin(2\pi 110t/f_s) + \sin(2\pi 250t/f_s), \qquad (30)$$

where the sample rate f_s equals 1000, $y(t)$ is composed of a low frequency component $200sin(2\pi 10t/f_s)$, a medium component $10sin(2\pi 110t/f_s)$, and a high component $sin(2\pi 250t/f_s)$. This signal is similar to the actual load in terms of the relative amplitude and frequency. A total of 3600 noisy data points $(t, \tilde{y}(t))$ are randomly generated:

$$\tilde{y}(t) = y(t) + \varepsilon(t), \qquad (31)$$

where $t \in [1, \ldots, 3600]$, and $\{\varepsilon(t)\}$ are independent and identically distributed Gaussian noises with zero mean and unit variance $N(0, 1)$. The first one-third of data points are used for training, the second one-third of data points for validation, and the last one-third of data points for test.

The WNNHKF is compared to the methods of persistence, linear AR, single NN without wavelet decomposition, and WNN. For all the methods, the relative increment transformation is not used for this example because $y(t)$ consists of three periodical sine

functions, and there is no need to use the transformation to make $\{y(t)\}$ stationary. As shown in Table 3-1, the numbers of hidden neurons of NNs are separately given, and these numbers are determined based on training, validation, and test processes in the three-way data split. To evaluate the accuracy, MAEs and SDs are calculated for 1- to 12-step-ahead predictions, and then they are separately averaged. The averaged MAE and averaged SD in Table 3-1 indicate that our method is better than the single NN. These results also indicate that the WNNHKF improves the WNN. For this example, MAPE is not used since $\{y(t)\}$ may have zero values.

Table 3-1. No. of Hidden Neurons, Averaged MAEs, and Averaged SDs Comparing the Results of WNNHKF to the Results of Persistence, Linear AR, Single NN, and WNN

	Persistence	Linear AR	Single NN	WNN	WNNHKF
No. of Neurons			16	15, 10, & 10 for LL, LH, & H	13, 10, & 10 for LL, LH, & H
Ave. MAE	51.58	2.30	2.06	1.68	1.46
Ave. SD	58.14	2.89	2.68	2.13	1.83

Example 2: Wavelet neural networks trained by hybrid Kalman filters are tested with ISO-NE's data. The training period is from January 1st, 2007 to December 31th, 2007, the validation is from January 1st, 2008 to June 30th, 2008, and the test is from July 1st, 2008 to December 31th, 2008. Five cases are presented. Cases 1-2 are for training and validation: Case 1 for the combination of $EKFNN_{LL}$ and $UKFNN_{LH, H}$ when compared to other combinations; and Case 2 for predictions with PIs. Cases 3-4 are for test: Case 3 for test results and PI evaluation; Case 4 for comparing the results of WNNHKF to the results of persistence, linear AR, ISO-NE's method, and WNN.

Case 1: The combination of EKFNN and UKFNN are examined with ISO-NE's load data. There are totally eight combinations of using EKFNN and UKFNNN for predicting three load components. To identify different strategies, the symbols LL, LH, and H are marked in subscripts. The validation results from 5- to 60-min-ahead predictions in Table 3-2 show that the combination of $EKFNN_{LL}$ and $UKFNN_{LH,\ H}$ produces the overall smallest MAPEs and SDs when compared to other seven strategies. This also supports the analysis in the beginning of Section 3.3 that the LL component has a near-linear relationship between input and output measurement, whereas LH and H components have highly nonlinear relationships. Here, the combination of $EKFNN_{LL}$, and $UKFNN_{H,\ LH}$ are treated as a nominal one and will be used for the rest of the testing.

Table 3-2. MAPEs (%) and SDs (MW) for Different Combinations of NNs Trained by Kalman Filter(s) for Individual Load Components

	$EKFNN_{H,LH,LL}$		$UKFNN_{H,LH,LL}$		$EKFNN_{H,\ LH}$ $UKFNN_{LL}$		$EKFNN_{H,\ LL}$ $UKFNN_{LH}$	
Min.	MAPE	SD	MAPE	SD	MAPE	SD	MAPE	SD
5	0.12	24.45	0.12	24.84	0.13	23.73	0.12	24.02
10	0.17	36.31	0.18	37.52	0.18	38.00	0.17	36.37
15	0.22	45.35	0.23	47.17	0.23	47.36	0.22	45.18
20	0.26	54.33	0.27	57.28	0.27	57.08	0.26	54.63
25	0.29	62.76	0.31	65.94	0.31	66.17	0.29	62.84
30	0.34	72.29	0.36	76.50	0.36	77.03	0.34	71.88
35	0.36	80.06	0.39	84.00	0.39	84.14	0.36	79.82
40	0.41	90.39	0.43	94.65	0.43	94.76	0.41	90.59
45	0.44	98.63	0.47	103.93	0.47	103.95	0.44	98.65
50	0.48	108.25	0.51	114.15	0.51	114.35	0.48	108.13
55	0.51	114.81	0.54	120.84	0.54	120.94	0.51	114.38
60	0.55	124.15	0.59	130.37	0.59	130.80	0.55	123.81

	$EKFNN_{LH, LL}$ $UKFNN_{H}$		$EKFNN_{H}$ $UKFNN_{LH, LL}$		$EKFNN_{LH}$ $UKFNN_{H, LL}$		$EKFNN_{LL}$ $UKFNN_{H, LH}$	
Min.	MAPE	SD	MAPE	SD	MAPE	SD	MAPE	SD
5	0.12	24.11	0.13	25.37	0.12	25.37	0.12	23.52
10	0.17	36.04	0.18	37.89	0.18	37.89	0.17	35.84
15	0.21	44.98	0.23	47.32	0.23	47.03	0.21	44.99
20	0.26	54.29	0.27	57.23	0.27	56.98	0.26	54.74
25	0.29	62.24	0.31	66.30	0.31	65.79	0.29	62.35
30	0.34	72.30	0.36	76.61	0.36	76.96	0.33	71.84
35	0.36	80.11	0.39	83.97	0.39	84.21	0.36	79.81
40	0.40	90.21	0.43	94.93	0.43	94.50	0.40	90.39
45	0.44	98.62	0.47	103.96	0.47	103.93	0.44	98.63
50	0.48	108.09	0.51	114.27	0.51	114.22	0.48	107.98
55	0.51	114.99	0.54	120.59	0.55	121.18	0.51	114.58
60	0.55	124.02	0.59	130.42	0.59	130.81	0.55	123.61

Case 2: The MAPEs, MAEs, SDs, ESDs, and one sigma coverage values as shown in Table 3-3 are calculated based on the validation data set. The first four measures gradually increase from 5- to 60-min-ahead forecasting results because the uncertainty expands as the forecasting step increases. Based on the observation, ESDs have values from 22 MW to 131 MW, and ISO-NE's system load data have values around 15000 MW. Since ESD values are much smaller than the system load magnitude, lower and upper bounds are always positive. For the case when the errors are not symmetric around estimates near zero, the bound can be truncated to a zero value if it is negative. Similar treatment can be found in Figure 2 of (Pinson and Kariniotakis, 2010). For the case when forecasted values are out-of-range, the load prediction after de-normalization can be clipped into a zero value if the prediction is negative or a historical maximum if the prediction is very high. The observation also shows that ESDs are very close to SDs. This corresponds to the analysis in Subsection 3.3.1 that the dynamic innovation

covariance is generally consistent with the covariance calculated based on static historical errors. Based on ESDs and predictions, the one sigma coverage values are calculated. Due to the heavy tails of errors (many large errors are related to peak hour load predictions), the coverage values for 5- to 60-min-ahead predictions have a range from 74% to 83% which are larger than the one sigma coverage rate of 68% under a Gaussian distribution. This indicates that PI estimates are reasonably accurate and conservative. The use of the Gaussian distribution is to be explained in Case 3.

Cases 1-2 above are for training and validation data sets, and the following Cases 3-4 are for the test data set.

Case 3: The five measures of the test data set in Table 3-4 are very close to the measures of the validation data set in Table 3-3. This indicates that WNNHKF parameters are properly selected. All the measures quantify forecasting accuracy in certain way, with the last two directly related to PIs. To further assess PI estimates, our standard-deviation-based PIs are evaluated and then compared to the empirical quantile-based PIs as follows.

Table 3-3. MAPEs (%), MAEs (MW), SDs (MW), ESDs (MW), and One Sigma Coverage (%) for WNNHKF Method (Based on Validation Data Set)

Min.	MAPE	MAE	SD	ESD	ONE SIGMA COVERAGE
5	0.12	17.22	23.52	22.79	74.27
10	0.17	25.48	35.84	35.60	77.52
15	0.21	31.64	44.99	49.29	81.16
20	0.26	37.70	54.74	55.62	79.30

25	0.29	42.98	62.35	61.19	77.93
30	0.33	50.12	71.84	75.70	80.75
35	0.36	54.16	79.81	81.43	80.84
40	0.40	60.38	90.39	97.32	82.78
45	0.44	65.98	98.63	102.60	81.85
50	0.48	72.12	107.98	107.60	80.75
55	0.51	76.52	114.58	112.53	80.40
60	0.55	82.76	123.61	130.08	82.33

Table 3-4. MAPEs (%), MAEs (MW), SDs (MW), ESDs (MW), and One Sigma Coverage (%) for WNNHKF Method (Based on Test Data Set)

Min.	MAPE	MAE	SD	ESD	ONE SIGMA COVERAGE
5	0.13	19.39	27.07	25.68	73.94
10	0.18	27.33	38.06	38.28	76.00
15	0.22	32.86	45.43	51.60	80.41
20	0.26	39.01	54.24	57.40	78.10
25	0.30	44.87	62.45	62.04	75.79
30	0.34	50.97	71.40	76.11	78.74
35	0.38	56.93	80.25	81.14	77.33
40	0.42	63.30	89.56	96.58	80.23
45	0.46	69.01	98.58	100.99	78.67
50	0.50	75.46	108.52	105.52	77.49
55	0.54	81.09	116.56	110.29	76.11
60	0.58	87.43	125.93	128.36	79.62

1) Evaluation of Standard-deviation-based PIs

As discussed in Subsection 3.4.2, the 5- to 60-min-ahead forecasting errors have heavier tails than a Gaussian distribution. Take 5-min errors from July to December 2008 for example. The Quantile-Quantile plot of the errors in Figure 3-5 clearly shows heavier tails than the Gaussian. After removing the tails below the 0.7th percentile or above the 99.3th percentile of the errors, the *p*-values of the Kolmogorov–Smirnov test, conducted in the two ways as described in Subsection 3.4.2, are both insignificant (>0.1). This indicates that the remaining errors have a zero mean Gaussian distribution. Furthermore, the ESD based on the entire sample of errors is close to the SD (in columns 4-5 of Tables 3.3-3.4). The ESD leads to an actual coverage rate of 74%, which is slightly larger than the one sigma coverage rate of 68% under a Gaussian distribution. Therefore, the distribution of the 5-min errors has heavier tails than a Gaussian distribution, but the total probability mass of the tails is very small (1.4%). Similarly, to make the Kolmogorov-Smirnov test insignificant for each of the other look-ahead times, as shown in Table 3-5, the total probability mass of tails is calculated as the fraction of errors that have been removed. Finally, the same conclusion is also made for 10- to 60-min forecasting results.

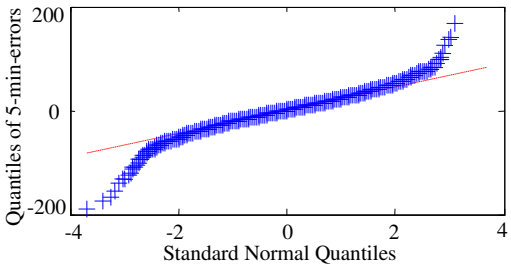

Figure 3-5. Quantile-Quantile plot of the 5 min-ahead forecasting errors versus the standard normal

Table 3-5. Total Probability Mass (%) of Tails of Error Removed to Make Kolmogorov–Smirnov Test Insignificant (p>0.1)

Min.	5	10	15	20	25	30
Total Probability Mass of Error Tails	1.40	2.80	5.78	4.16	5.16	5.56
Min.	35	40	45	50	55	60
Total Probability Mass of Error Tails	5.78	3.80	5.10	6.82	5.74	5.82

In view of the above distribution analysis, to evaluate PI estimates, three comparisons are conducted. First, as shown in columns 4-5 of Table 3-4, the ESDs are quite close to the SDs for 5- to 60-minute outs. Second, as shown in column 6 of Table 3-4, the one sigma coverage rates for 5- to 60-min-ahead predictions range from 73% to 80% which are larger than 68% under the standard Gaussian distribution. Third, consider WNNHKF 5-min outs from July to December 2008 for example. To achieve the 90% coverage rate, the amount of the ESD is found to be 1.52, which is slightly smaller than 1.64 under the standard Gaussian distribution. The last two comparisons indicate the ESD is conservative. The same conclusion is also made for 10- to 60-min outs and for different coverage rates, i.e., 10%, 20%, …, 90%, as shown in Table 3-6. To further illustrate the conclusion, we graph the amount of ESDs as a function of coverage rates ranging from 10% to 90% for each look-ahead time, and compare it to the amount of sigmas under the standard Gaussian graphed in the same way. As shown in Figure 3-6,

the curve for the ESD is always slightly below the curve for the standard Gaussian, indicating conservative PIs. Based on these, it can be concluded that the PI estimates for coverage rates up to 90% are reasonably accurate and conservative.

Table 3-6. Amount of ESD to Achieve Almost the Same Coverage Rates

	COVERAGE RATE								
Min.	10%	20%	30%	40%	50%	60%	70%	80%	90%
5	0.11	0.22	0.33	0.45	0.58	0.73	0.90	1.13	1.52
10	0.09	0.19	0.30	0.42	0.54	0.69	0.86	1.08	1.42
15	0.08	0.16	0.26	0.37	0.48	0.61	0.77	0.97	1.30
20	0.08	0.18	0.29	0.39	0.51	0.64	0.80	1.02	1.39
25	0.09	0.20	0.30	0.41	0.53	0.68	0.86	1.09	1.49
30	0.08	0.17	0.28	0.38	0.49	0.63	0.79	1.01	1.37
35	0.09	0.18	0.29	0.40	0.51	0.66	0.82	1.05	1.42
40	0.09	0.18	0.27	0.37	0.49	0.61	0.76	0.97	1.31
45	0.09	0.18	0.28	0.39	0.49	0.63	0.78	1.01	1.38
50	0.09	0.19	0.29	0.39	0.52	0.65	0.83	1.05	1.44
55	0.09	0.19	0.30	0.41	0.53	0.67	0.84	1.08	1.49
60	0.09	0.18	0.27	0.38	0.49	0.63	0.78	0.99	1.38
	COVERAGE RATE								
Min.	91%	92%	93%	94%	95%	96%	97%	98%	99%
5	1.57	1.63	1.70	1.77	1.86	1.94	2.12	2.25	2.45
10	1.48	1.55	1.62	1.70	1.80	1.89	2.03	2.25	2.49
15	1.35	1.39	1.47	1.56	1.65	1.73	1.85	1.99	2.20
20	1.44	1.51	1.59	1.66	1.75	1.84	1.97	2.17	2.38
25	1.54	1.60	1.67	1.73	1.84	1.98	2.13	2.34	2.62
30	1.42	1.49	1.57	1.65	1.72	1.82	1.98	2.12	2.41
35	1.48	1.56	1.64	1.73	1.81	1.92	2.11	2.30	2.59
40	1.35	1.42	1.50	1.59	1.68	1.83	2.04	2.25	2.45
45	1.43	1.49	1.56	1.68	1.78	1.92	2.16	2.38	2.64
50	1.51	1.58	1.68	1.79	1.89	2.02	2.24	2.46	2.82
55	1.55	1.63	1.70	1.77	1.94	2.09	2.30	2.53	2.89
60	1.44	1.51	1.59	1.68	1.78	1.90	2.07	2.37	2.68

To explore further, the PI estimates for coverage rates above 90% are investigated. We have seen from Figure 3-5 that forecasting errors can significantly deviate from the Gaussian distribution as they become more extreme. To attain very high coverage rates, a large number of extreme errors have to be accounted for. To assess the effects of these non-Gaussian extreme errors, curves similar to those in Figure 3-6 are graphed, but with coverage rates ranging from 91% to 99%, as shown in Table 3-6. Figure 3-7 shows that for coverage rates up to 95%, the amounts of ESDs for 5 to 60-min outs are slightly lower than those derived from the Gaussian distribution, indicating the PI estimates are still accurate and conservative. The result is also consistent with the observation from Table 3-5 that the total probability mass of the tails ranges from 1.40% to 6.82% for 5 to 60-min outs. For coverage rates higher than 95%, the curves of the ESD for some look-ahead times (i.e., 5- to 20-min-ahead and 30- to 40-min-ahead times) are still below the curve for the Gaussian distribution, indicating conservative PIs. On the other hand, generally speaking, for larger look-ahead times, the curves of the ESD are above the curve for the Gaussian distribution, indicating large errors. This is consistent with the fact that as look-ahead time increases, data uncertainty increases as well.

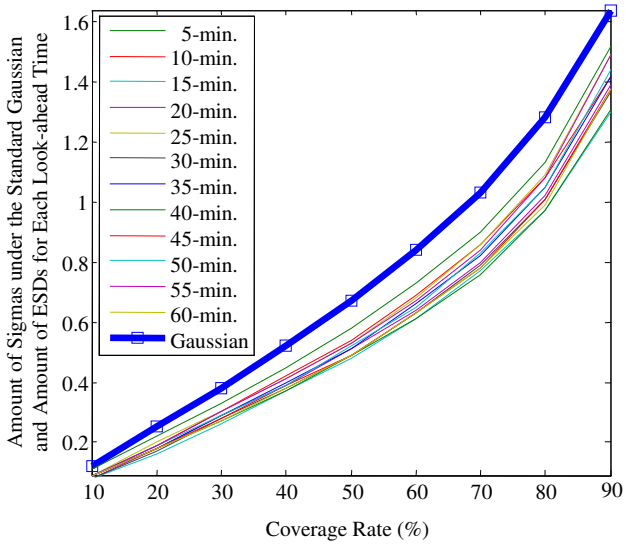

Figure 3-6. The amount of ESDs as a function of coverage rates ranging from 10% to 90% for each look-ahead time when compared with the amount of sigmas under the standard Gaussian

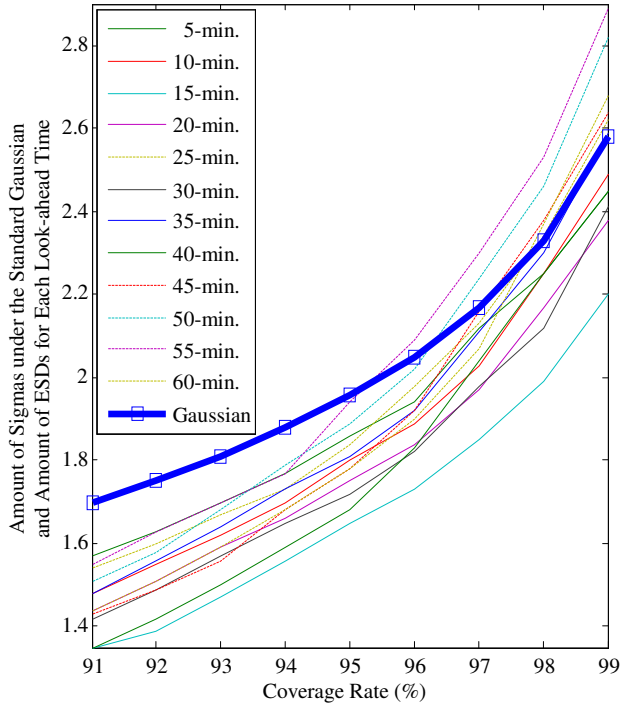

Figure 3-7. The amount of ESDs as a function of coverage rates ranging from 91% to 99% for each look-ahead time when compared with the amount of sigmas under the standard Gaussian

2) Standard-deviation-based PIs versus Quantile-based PIs

The standard-deviation-based PIs are further evaluated by comparison to the empirical quantile-based PIs which are constructed for nominal coverage rates of 10%, 20%, ..., 90%. To construct empirical quantile-based PIs, consider WNNHKF 5-min-ahead forecasting errors for example. At time t, historical 5-min errors (actual minus predicted load of 5-min-ahead) before time t are collected. For a nominal coverage rate 1-α, e.g., α =0.1, the 5^{th} and 95^{th} percentiles of the errors are calculated. The 90% prediction interval for time t is obtained by adding the 5^{th} and 95^{th} percentiles to the predicted load. For our testing, the errors from July 1^{st}, 2008 to November 30^{th}, 2008 are used to construct the quantile-based PI for t = *00:05am* on December 1^{st}, 2008. When the error at *00:05 am* becomes available, the new error and previous errors are then combined to construct the prediction interval for t = *00:10 am*, and so on. To quantify forecasting accuracy, this process is repeated for all the data collected until the end of December. The result shows that the empirical quantile-based PIs of 90% nominal coverage rate cover 90.44% of the actual load data, indicating the empirical quantile-based PIs are accurate. The same steps are taken for 10- to 60-min forecasting results and for different nominal coverage rates, ranging from 10% to 90%, and similar conclusions are obtained as shown in Table 3-7.

Table 3-7. Actual Coverage Rates (%) of Empirical Quantile-based PIs for Different Nominal Coverage Rates

	NOMINAL COVERAGE RATE								
Min.	10%	20%	30%	40%	50%	60%	70%	80%	90%
5	10.77	18.43	27.19	38.22	48.05	57.80	67.56	78.73	90.44
10	9.56	19.52	31.36	40.65	49.80	59.62	69.04	77.93	90.44
15	11.17	19.78	29.74	37.95	47.38	57.74	66.89	77.52	88.96
20	10.23	20.86	31.22	40.78	48.32	58.82	67.43	77.52	88.69
25	11.04	21.27	30.82	39.70	49.80	57.34	66.49	75.64	87.21
30	10.23	20.73	30.82	40.65	48.99	58.28	67.97	77.66	88.29
35	9.69	19.78	29.74	39.17	47.78	57.60	68.37	76.72	87.35
40	8.34	18.57	30.01	38.22	47.78	57.87	65.95	76.04	86.94
45	9.42	20.18	29.61	38.63	48.45	57.60	67.03	76.58	87.08
50	10.36	18.44	29.21	39.03	47.91	56.39	66.76	76.58	86.68
55	9.69	19.11	27.73	36.88	48.32	57.74	67.29	76.31	86.94
60	8.88	18.44	26.92	36.47	46.84	56.39	67.03	75.91	87.08

The standard-deviation-based PIs are derived from dynamic innovation covariance of Kalman filters, whereas the empirical quantile-based PIs are derived from quasistatic historical errors. To compare these two types of PIs on an equal footing, the widths of the PIs under the same actual coverage rates are compared. Again, consider WNNHKF 5-min outs for December 2008 for example. Under the 90% nominal coverage rate, the empirical quantile-based PIs have an actual coverage rate of 90.44% with an average width of 81.17MW. To achieve the same actual coverage rate, the width of standard-deviation-based PIs is found to be 1.47 × 2ESD with an average width of 76.28 MW. The comparison indicates that under the same actual coverage rate, the standard-

deviation-based PIs are generally narrower than the empirical quantile-based PIs. This result is consistent with the dynamic nature of the innovation covariance produced by Kaman filters as explained in Subsection 3.3.1. The same steps are taken for 10- to 60-min-ahead forecasting results and for different nominal coverage rates ranging from 10% to 90%, and similar results can be obtained from Tables 3-8 and 3-9.

Table 3-8. Widths (MW) of Empirical Quantile-based PIs for Different Nominal Coverage Rates as Shown in Table 3-7

	NOMINAL COVERAGE RATE								
Min.	10%	20%	30%	40%	50%	60%	70%	80%	90%
5	5.81	11.19	16.81	23.07	29.75	37.20	46.48	59.13	81.17
10	7.08	15.20	23.13	31.77	41.34	52.86	65.43	83.63	116.90
15	8.29	17.29	27.00	36.81	49.48	62.82	79.43	102.97	143.54
20	9.80	20.55	32.61	44.80	58.12	73.52	93.58	119.83	167.01
25	11.85	24.86	36.87	50.18	65.20	83.44	107.20	136.81	193.64
30	12.94	26.67	42.51	57.30	74.05	94.57	120.14	157.65	220.65
35	14.46	29.67	46.42	62.86	81.87	106.65	134.15	174.24	244.98
40	16.29	34.40	51.92	70.34	91.84	116.61	145.71	188.65	266.49
45	17.57	36.85	56.36	74.89	97.33	126.86	157.81	206.28	293.29
50	19.38	39.24	59.46	79.78	106.03	138.18	174.09	227.45	321.83
55	20.01	41.28	64.10	87.28	116.03	146.94	186.87	243.40	342.05
60	22.73	44.81	68.47	94.35	122.52	159.25	202.75	260.95	369.07

Table 3-9. Widths (MW) of Standard-Deviation-based PIs Achieving the Same Actual Coverage Rates as Shown in Table 3-7

	NOMINAL COVERAGE RATE								
Min.	10%	20%	30%	40%	50%	60%	70%	80%	90%
5	5.71	11.42	17.12	22.83	29.58	36.33	45.67	58.12	76.28
10	6.94	15.43	23.91	33.17	42.43	52.45	64.79	79.45	110.31
15	8.32	16.63	28.07	36.38	49.90	62.37	81.09	99.80	133.06
20	9.26	18.51	31.23	40.49	55.53	69.42	90.24	111.07	148.09
25	9.99	19.98	33.72	43.71	59.95	74.94	97.42	119.90	159.87
30	12.26	24.51	41.37	53.63	73.54	91.93	119.51	147.09	196.12
35	13.05	26.11	44.06	57.11	78.32	97.90	127.27	156.64	208.86
40	15.52	31.05	52.39	67.91	93.13	116.42	141.64	184.33	242.54
45	16.23	32.48	54.81	71.05	97.43	121.79	148.18	192.84	253.73
50	16.97	33.95	57.29	74.26	101.85	127.31	154.89	201.57	265.23
55	17.71	35.43	59.78	77.49	106.28	132.85	161.63	210.34	276.76
60	20.60	41.20	66.95	90.13	123.61	154.51	187.99	244.64	321.89

Case 4: Results of our WNNHKF method are compared to the results of persistence, linear AR model, ISO-NE's method in (Shamsollahi et al., 2001), and WNN method of (Guan et al., 2013) reviewed in Section 3.2.1, based on the ISO-NE's data set. The forecasting period for comparison is from July 1st, 2008 to July 31th, 2008 because ISO-NE only provided results of this period to us. MAPEs in Table 3-10 show that the results of our method are better than the results of persistence, linear AR model, and ISO-NE's method. The same conclusion is also made from MAEs. Furthermore, our WNNHKF method improves the WNN for 10- to 60-min-ahead predictions, but doesn't perform as well as the WNN for 5-min-ahead predictions. This is because the relationship between input and output measurement for an NN does not appear to be very

nonlinear based on observation, and the UKFNN may not work as well as the standard NN for 5-min-ahead predictions. The same conclusion is made for winter and spring seasons (December 2008 to May 2009) when the performance of WNNHKF and WNN are compared. For the same reason, the WNNHKF doesn't perform as well as the WNN for fall season (September to November 2008).

Table 3-10. MAPEs (%) Comparing the Results of WNNHKF to the Results of Persistence, Linear AR Model, ISO-NE's Method, and WNN

Min.	Persistence	Linear AR	ISO-NE's Method	WNN	WNNHKF
5	0.38	0.16	0.26	0.08	0.12
10	0.74	0.22	0.30	0.13	0.13
15	1.10	0.32	0.34	0.16	0.15
20	1.46	0.44	0.38	0.20	0.16
25	1.82	0.57	0.43	0.23	0.18
30	2.18	0.71	0.48	0.27	0.23
35	2.53	0.85	0.53	0.31	0.26
40	2.89	1.01	0.60	0.35	0.33
45	3.24	1.17	0.64	0.38	0.37
50	3.59	1.35	0.70	0.42	0.36
55	3.94	1.54	0.75	0.45	0.40
60	4.29	1.73	0.81	0.49	0.47

3.6 Conclusion

This paper presents a method of wavelet neural networks trained by hybrid Kalman filters. Based on data analysis, an EKFNN is used to capture the near-linear relationship between the LL input and output measurement for an NN, and two UKFNNs are used to capture the highly nonlinear relationships for LH and H load components. By replacing

the first-order back propagation algorithm with the second-order Kalman-type algorithms, the dynamic innovation covariance can be obtained for PI estimates. Consequently, the estimated standard deviation, which is derived based on the nonlinear transformation of WNNHKF, is close to the sample standard deviation. To evaluate PIs, the forecasting errors are demonstrated to have heavier tails than a Gaussian distribution. For the forecasting errors, both the one sigma coverage and the amount of the estimated standard deviations needed to achieve a given coverage rate are close to the ones under the standard Gaussian distribution. Numerical testing results based on ISO-NE's data show that the WNNHKF provides the overall best predictions with accurate and conservative PI estimates.

4. Summary and Future Research

4.1 Summary

In the previous chapters, the wavelet neural network based models have been presented for very short term load forecasting with prediction interval estimates. Major features of the methods are highlighted below:

The spike filtering methods effectively remove spikes in real-time.

The WNN method can capture the load components at different frequencies. Since the data input-output measurement relation changes over time due to human behavior, a single structure of wavelet neural networks cannot capture well. Twelve dedicated wavelet neural networks, based on test results, are used to perform moving forecasts every five minutes over an hour.

By replacing the first-order back propagation algorithm with the second-order Kalman-type algorithms, the dynamic innovation covariance can be obtained for prediction interval estimates. EKFNN can capture the near-linear relationship between the LL input and output measurement for an NN, and UKFNNs can capture the highly nonlinear relationships for LH and H load components.

To evaluate PIs, the forecasting errors are demonstrated to have heavier tails than a Gaussian distribution. For the forecasting errors, both the one sigma coverage and the amount of the estimated standard deviations needed to achieve a given coverage rate are close to the ones under the standard Gaussian distribution. Derived prediction intervals are accurate and conservative.

Results of the presented methods are better than the results of persistence, linear AR model, and ISO-NE's method. WNN is based on the 1-order back propagation without estimating prediction intervals. While WNNHKF is based on the 2-order Kalman type algorithms, where the by-product dynamic covariance can be easily produced. WNNHKF improves WNN for most of the minute outs.

4.2 Future Research Directions

With the smart grid initiative, the generation and load patterns, and more importantly, the way people use electricity, will be fundamentally changed. With intermittent renewable generation, advanced metering infrastructure, dynamic pricing, intelligent appliances and HVAC equipment, micro grids, and hybrid plug-in vehicles, etc., load forecasting five years from now will be quite different from today.

Recently, load forecasting at the distribution level is becoming important for the planning and operation of distribution systems. Such forecasting is more difficult than forecasting the total system load because of the large number of substations, and the complicated spatial and temporal correlations of their load. The presence of significant distributed energy resources and demand response in the future will add other levels of complexity. To address the above difficulties, the method of wavelet neural network trained by Kamlam filters can be extended to load forecasting at the distribution level as well as demand response data, i.e., load and price, and the distributed energy resources, e.g., wind generation.

5. **Bibliography**

1. A. P. Alves da Silva and L. S. Moulin, " Confidence intervals for neural network based short-term load forecasting," *IEEE Transaction on Power Systems*, Vol. 15, No. 4, pp. 1191-1196, 2000.
2. Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, "Estimation with applications to rracking and navigation: algorithms and software for information extraction," *J. Wiley and Sons*, 2001.
3. D. Benaouda, F. Murtagh, J. L. Starck, and O. Renaud, "Wavelet-based nonlinear multi-scale decomposition model for electricity load forecasting," *Neurocomputing*, vol. 70, pp. 139-154, 2006.
4. R. J. Bessa, V. Miranda, and J. Gama, "Entropy and correntropy against minimum square error in offline and online three-day ahead wind power forecasting," *IEEE Transactions on Power Systems*, vol. 24, no. 4, pp. 1657-1666, November 2009.
5. W. Charytoniuk and M. S. Chen, "Very short-term load forecasting using artificial neural networks," *IEEE Transactions on Power Systems*, vol. 15, no. 1, pp. 263-268, February 2000.
6. W. Charytoniuk, M. S. Chen, P. Kotas, and P. Van Olinda, "Demand forecasting in power distribution systems using nonparametric. probability density estimation," *IEEE Transactions on Power Systems*, Vol. 14, No. 4, pp. 1200-1206, November 1999.
7. C. Charytoniuk and J. Niebrzydowski, "Confidence interval construction for load forecast," *Electric Power Systems Research*, pp. 48, 97-103, 1998.

8. Y. Chen, P. B. Luh, C. Guan, Y. G. Zhao, L. D. Michel, M. A. Coolbeth, P. B. Friedland, and S. J. Rourke, "Short-term load forecasting: similar day-based wavelet neural networks," *IEEE Transactions on Power Systems*, vol. 25, no. 1, pp. 322-330, February 2010.
9. G. Chryssolouris, M. Lee, and A. Ramsey, "Confidence interval prediction for neural network models," *IEEE Transactions on Neural Networks*, Vol. 7, No. 1, pp. 229-232, 1996.
10. L. C. M. de Andrade and I. N. da Silva, "Very short-term load forecasting using a hybrid neuro-fuzzy approach," *2010 Eleventh Brazilian Symposium on Neural Networks*, Sao Carlos, Brazil, October 2010.
11. B. Fox, D. Flynn, L. Bryans, N. Jenkins, D. Milborrow, M. O'Malley, R. Watson, and O. Anaya-Lara, "Wind power integration connection and system operational aspects," *IET Power and Energy Series*, 2007.
12. L. C. M. de Andrade and I. N. da Silva, "Using intelligent system approach for very short-term load forecasting purposes," *2010 IEEE International Energy Conference and Exhibition*, Manama, Bahrain, December 2010.
13. S. Fan and R. J. Hyndman, "Short-term load forecasting based on a semi-parametric additive model," *IEEE Transactions on Power Systems*, vol. 27, no. 1, pp. 134-141, February 2012.
14. J. N. Fidalgo and J. A. Peças Lopes, "Load forecasting performance enhancement when facing anomalous events," *IEEE Transactions on Power Systems*, vol. 20, no. 1, pp. 408-415, February 2005.

15. C. Guan and P. B. Luh, "Derivation of the padding length for Daubechies wavelets," working paper, referred upon the request, 2010.
16. C. Guan, P. B. Luh, M. A. Coolbeth, Y. Zhao, L. D. Michel, Y. Chen, C. J. Manville, P. B. Friedland, and S. J. Rourke, "Very short-term load forecasting: multilevel wavelet neural networks with data pre-filtering," *Proceedings of the IEEE Power and Energy Society 2009 General Meeting*, Calgary, Alberta, Canada, July 2009.
17. C. Guan, P. B. Luh, L. D. Michel, M. A. Coolbeth, and P. B. Friedland, "Hybrid Kalman algorithms for very short-term load forecasting and prediction interval estimation," *Proceedings of the IEEE Power and Energy Society 2010 General Meeting*, Minneapolis, Minnesota, 2010.
18. C. Guan, P. B. Luh, L. D. Michel, Y. Wang, and P. B. Friedland, "Very short-term load forecasting: wavelet neural networks with data pre-filtering," *IEEE Transactions on Power Systems*, vol. 28, no. 1, pp. 30-41, February 2013.
19. S. Haykin, "Neural networks and learning machines, third edition," *Prentice Hall*, 2009.
20. M. Herrera, L. Torgo, J. Izquierdo, and R. Pérez-García, "Predictive models for forecasting hourly urban water demand," *Journal of Hydrology*, vol. 387, pp. 141-150, 2010.
21. R. J. Hyndman, "Another look at forecast accuracy metrics for intermittent demand," *Foresight: the International Journal of Applied Forecasting*, Issue 4, pp. 43-46, June 2006.
22. R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International Journal of Forecasting*, vol. 22, pp. 679-688, 2006.

23. S. J. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte, "A new approach for filtering nonlinear systems," *Proceedings of the American Control Conference*, pp. 1628-1632, 1995.

24. S. Kawauchi, H. Sugihara, and H. Sasaki, "Development of very short-term load forecasting based on chaos theory," *Electrical Engineering in Japan*, vol. 148, no. 2, pp. 55-63, 2004.

25. I. Koprinska, R. Sood, and V. Agelidis, "Variable selection for five-minute ahead electricity load forecasting," *20th International Conference on Pattern Recognition*, Istanbul, Turkey, August 2010.

26. K. Liu, S. Subbarayan, R. R. Shoults, M. T. Manry, C. Kwan, F. L. Lewis, and J. Naccarino, "Comparison of very short-term load forecasting techniques," *IEEE Transactions on Power Systems*, vol. 11, no. 2, pp. 877-882, May 1996.

27. W. Liu, P. P. Pokharel, and J. C. Principe, "Correntropy: properties and applications in non-Gaussian signal processing," *IEEE Transactions on Signal Processing*, vol. 55, no. 11, pp. 5286-5298, November 2007.

28. J. Lu, X. Zhang, and W. Sun, "A real-time adaptive forecasting algorithm for electric power load," *2005 IEEE PES Transmission and Distribution Conference and Exposition: Asia and Pacific,* Dalian, China, 2005.

29. D. Luo and H. He, "A shape similarity criterion based curve fitting algorithm and its application in ultra-short-term load forecasting," *Power System Technology*, vol. 31, no. 21, pp. 81-84, November 2007.

30. S. G. Mallat, "A wavelet tour of signal processing: the sparse way, third edition," *Academic Press*, 2009.

31. S. K. Mitra, "Digital signal processing: a computer-based approach, third edition," *McGraw Hill Inc.*, New York, 2006.
32. G. Papadopoulos, P. J. Edwards, and A. F. Murray, "Confidence estimation methods for neural networks: a practical comparison," *IEEE Transaction on Neural Networks*, vol. 12, no. 6, pp. 1278-1287, 2001.
33. P. Pinson and G. Kariniotakis, "Conditional prediction intervals of wind power generation," *IEEE Transactions on Power Systems*, vol. 25, no. 4, pp. 1845-1856, November 2010.
34. G. V. Puskorius and L. A. Feldkamp, "Decoupled extended Kalman filter training of feedforward layered networks," *International Joint Conference on Neural Networks*, Dearborn, Michigan, July 1991.
35. A. J. Rocha Reis and A. P. Alves da Silva, "Feature extraction via multi-resolution analysis for short-term load forecasting," *IEEE Transactions on Power Systems*, vol. 20, no. 1, pp. 189-198, February 2005.
36. B. D. Ripley, "Neural networks and pattern recognition," *Cambridge University Press*, 1996.
37. A. Setiawan, I. Koprinska, and V. G. Agelidis, "Very short-term electricity load demand forecasting using support vector regression," *2009 International Joint Conference on Neural Networks,* Atlanta, GA, June 2009.
38. P. Shamsollahi, K. W. Cheung, Q. Chen, and E. H. Germain, "A neural network based very short-term load forecaster for the interim ISO New England electricity market system," *Innovative Computing for Power - Electric Energy Meets the*

Market: 22nd IEEE Power Engineering Society International Conference on Power Industry Computer Applications, Sydney, Australia, May 2001.

39. S. Singhal and L. Wu, "Training feed-forward networks with the extended Kalman algorithm," *International Conference on Acoustics, Speech, and Signal Processing*, Morristown, New Jersey, May 1989.
40. S. W. Smith, "Scientist and engineer's guide to digital signal processing, second edition," *California Technical Publishing*, 1999.
41. G. Strang and T. Nguyen, "Wavelets and filter banks, second edition," *Wellesley-Cambridge Pres*s, Wellesley, Massachusetts, 1997.
42. Stat Trek, "AP statistics: boxplots (aka, Box and Whisker plots)," stattrek.com/ap-statistics-1/boxplot.aspx.
43. J. W. Taylor, "An evaluation of methods for very short-term load forecasting using minute-by-minute British data," *International Journal of Forecasting*, vol. 24, pp. 645-658, 2008.
44. The weather world 2010 project, "Weather forecasting," ww2010.atmos.uiuc.edu/%28Gl%29/guides/mtr/fcst/mth/oth.rxml.
45. D. J. Trudnowski and W. L. Mcreynolds, "Real-time very short-term load prediction for power system automatic generation control," *IEEE Transactions on Control Systems Technology*, vol. 9, no. 2, pp. 254-260, March 2001.
46. E. A. Wan and R. Van Der Merwe, "The Unscented Kalman Filter for Nonlinear Estimation," *Proceedings of the IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium*, Lake Louise, Alberta, Canada, 2000.

47. F. Wang, K. Xie, E. Yu, G. Liu, and M. Wang, "A simple and effective ultra-short term load forecasting method," *Power System Technology*, vol. 20, no. 3, pp. 41-43, 48, March 1996.

48. W. A. Wright, "Bayesian approach to neural network modeling with input uncertainty," *IEEE Transaction on Neural Networks*, Vol. 10, No. 6, pp. 1261-1270, November 1999.

49. K. Xie, F. Wang, and E. Yu, "Very short-term load forecasting by Kalman filter algorithm," *Proceedings of the Chinese Society for Electrical Engineering*, vol. 16, no. 4, pp. 245-249, July 1996.

50. H. Yang, H. Ye, G. Wang, J. Khan, and T. Hu, "Fuzzy neural very short-term load forecasting based on chaotic dynamics reconstruction," *Chaos, Solitons & Fractals*, vol. 29, pp. 462-469, 2006.

51. Z. Yang, G. Tang, Y. Song, and R. Cao, "Improved cluster analysis based ultra-short term load forecasting method," *Automation of Electric Power System*, vol. 29, no. 24, pp. 83-86, December 2005.

52. L. Zhang and P. B. Luh, "Neural network based market clearing price prediction and prediction interval estimation with an improved extended Kalman filter method," *IEEE Transaction on Power Systems,* Vol. 20, No. 1, pp. 59-66, February 2005.

53. L. Zhang, P. B. Luh, and K. Kasiviswanathan, "Energy clearing price prediction and confidence interval estimation with cascaded neural networks," *IEEE Transaction on Power Systems*, Vol. 18, No. 1, pp. 99-105, February 2003.

54. Y. Zhao, P. B. Luh, C. Bomgardner, and G. H. Beerel, "Short-term load forecasting: multilevel wavelet neural networks with holiday corrections," *Proceedings of the*

IEEE Power and Energy Society 2009 General Meeting, Calgary, Alberta, Canada, July 2009.

55. J. Zhou, B. Zhang, J. Shang, J. Yao, and M. Cheng, "Very short-term load forecast based on multi-sample extrapolation and error analysis," *Electric Power Automation Equipment*, vol. 25, no. 2, pp. 15-21, February 2005.